VisualFoundation 12.0

User's Guide

Updated: 11/7/24

VisualFoundation 12.0 User's Guide

Table of Contents

1 Introduction

1.1 Welcome to VisualFoundation 12.0

A New Commercial Version is Available.

IES has upgraded VisualFoundation 12.0. The latest release can be found on our webstie at: www.iesweb.com/downloads

VisualFoundation will help you design complex mat footings, pile caps, grade beam systems, and combined footings. It automates much of the work involved in finite element modeling within a more complex and flexible tool like VisualAnalysis. VisualFoundation will take care of FEA details, while you specify the locations, sizes and configuration of your structure. The software will analyze your footing slab to determine moments, shears, displacements and bearing pressures. VisualFoundation will then help you check to make sure the footing demand/capacity ratios (unity checks) pass, and it will tell you about the steel requirements for the slab. It will also help with the design of grade beams, column-piers, and piles.

Getting Started

- Use **File | Open** Example to see sample projects.
- Feature List
- Program Layout
- Upgrade Guide (What's New?)
- FAQ Answers at iesweb.com for business, licensing, installation issues.

Help Notation

- Menu items appear like this: **File | New**.
- Keystrokes or mouse commands appear like this: ***Shift+Click***.

Disclaimer

VisualFoundation is a proprietary computer program of Integrated Engineering Software (IES, Inc.) of Bozeman, MT. This product is intended for use by licensed, practicing engineers who are educated in structural engineering, students in this field, and related professionals (e.g. Architects, Building Inspectors, Mechanical Engineers, etc.). Although every effort has been made to ensure the accuracy of this program and its documentation, IES, Inc. does not accept responsibility for any mistake, error, or misrepresentation in, or as a result of, the usage of this program and its documentation. (Though we will make every effort to insure that problems that we can correct are dealt with promptly.) The results obtained from the use of this program should not be substituted for sound engineering judgment.

License and Copy Restrictions

By installing VisualFoundation on your computer, you become a registered user of the software. The VisualFoundation program is the copyrighted property of IES, Inc. and is provided for the exclusive use of each licensee. You may copy the program for backup purposes and you may install it on any computer allowed in the license agreement. Distributing the program to coworkers, friends, or duplicating if for other distribution violates the copyright laws of the United States. Future enhancements and technical support depend on your cooperation in this regard. Additional licenses and/or copies of VisualFoundation may be purchased directly from IES, Inc.

IES, Inc.

Integrated Engineering Software, Inc.
519 E. Babcock St.
Bozeman, MT 59718
Sales or Licensing: 406-586-8988, sales@iesweb.com
Technical Support: support@iesweb.com

1.2 Features

General

- Simple, standard Windows interface for easy navigation
- Unlimited Undo & Redo commands
- Work in any unit system, perform math on input, use custom unit 'styles'
- Program is self-documenting with tooltips on commands and input parameters
- Numerous preference settings for better defaults
- Drive the program with the Command Line
- Run External Scripts to automate common tasks and more
- Free training videos provided for learning efficiency
- Free technical support email with fast, friendly turnaround

Modeling

- Quick-start with typical foundation geometries
- Import CAD complex and detailed boundaries
- Import project from VisualAnalysis, with pier and wall locations as well as loads
- Sketch multiple boundaries graphically with rectangular or polar gird lines
- Work with column-lines to label and modify footings quickly
- Multiple-thickness footings with different soil conditions under each footing boundary
- Use Grade beams and walls to add stiffness to foundations
- Model piers (pedestals) for columns
- Add piles for additional foundation support (steel, wood, or concrete shapes)
- Specify lateral sliding resistance from soil or piles

Loading

- Model objects can be loaded in multiple service load cases
- Automatic building code load combinations are available
- Includes IBC, ASCE 7, and NBC Load Combinations
- Customizable building code combinations (see Load Case Manager)
- Create custom load combinations in any project
- Apply loads to entire foundation or rectangular, circular, tube or ring shapes
- Apply loads through column-piers, onto grade beams or walls
- Copy and paste loads to objects
- Copy and scale loads to other load cases

Analysis

- FEA model is constructed automatically by the software
- Automated "background" analysis is fast
- Plate elements use a "thick-plate" formulation for accurate shear results
- Nonlinear iteration for compression-only spring supports
- Advanced error-checking and reporting
- Export project to VisualAnalysis for more sophisticated or dynamic analysis

Design

- **ACI 318-19, ACI 318-14, CSA A23.3:19, CSA A23.3-14** concrete specifications
- **AISC 360-16, AISC 360-10, AISC 360-05 (ASD or LRFD), or CSA S16:19, CSA S16-14** steel specifications (piles)
- **NDS 2018 ASD or LRFD** wood specification (piles)
- Determination of concrete rebar demands
- Code checks for concrete foundation slabs with your specified primary rebar
- Code checks for concrete piers and piles including your rebar details
- Built in rebar in USA and metric sizes
- Intermediate values shown in reports for easier plan-check review

Reporting

- Quick Full Report includes graphics and all details, with options
- Custom reporting to include just the information you need
- Print Preview mode while working with reports
- Paste any graphics into your report
- Customizable page margins, fonts, colors
- Use your own company logo in report page headers
- Print to any printer including PDF
- Export to text clipboard or save to other formats like .xlsx

Limitations

- Does not **detail** the slab rebar placement (i.e., no bar lengths, cutoffs, bends, etc.)
- Does not produce structural **drawings**
- No soil-structure analysis of pile behavior
- No design for **walls** that are part of a foundation (See QuickConcreteWall, QuickMasonry or VisualAnalysis)
- Cannot model a system of **disconnected** footings.

Be a Squeaky Wheel

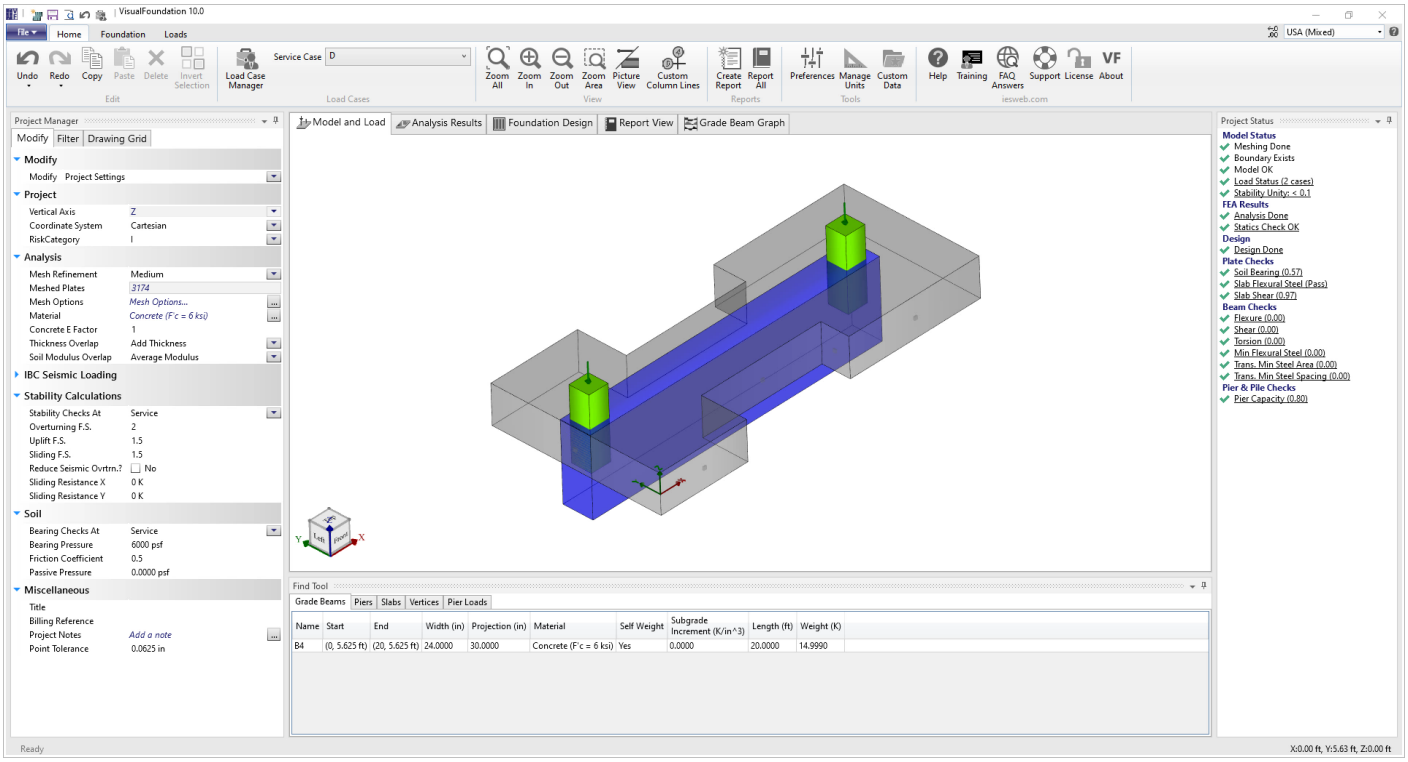
If you need a new feature, please let us know! We are always looking for ways to improve products in ways that you desire. See Technical Support.

1.3 Program Layout

The best way to learn VisualFoundation is to use and explore the program to get to know what is available under each button or menu. Several Tutorial Videos are also available which explain many features of the software.

Screen Layout

The image below introduces the program terminology used in this help file and the training videos. Panels may be resized by dragging their dividers or repositioned by dragging their title bars or right-clicking on the title. Use the "pushpin" icon to collapse panels temporarily to gain more space for working. Hold the mouse pointer over the screen image below for information about each area of the program.



Title Bar

The title bar displays the version of VisualFoundation and the file name of the project. Also, there are helpful buttons in addition to the Windows system buttons.

Main Menu / Toolbar

The main menu, Toolbar, or Ribbon, contains various commands to direct VisualFoundation. Each is organized within a group to help locate them quickly. Each command has a description which appears when the mouse pointer is hovered over it. Many have hot-key shortcuts.

Project Manager

The Project Manager provides immediate access to frequent operations in VisualFoundation. This tool is docked on the left side of the window by default and displays various tabs depending on the active window. This window can be docked on the left or made to float independently if more space is needed to work. Alternatively, drag the side border to make it wider or narrower.

- The **Modify** tabs are used to change the project settings or the properties of selected objects in the Model and Load View or the Foundation Design View.
- The **Filter** tab is used to control what is shown or hidden in the active view.

- The **Drawing Grid** tab is used to control the Sketch Grid to aid in drawing models in the Model and Load View.
- The **Result** tab replaces the Modify tab when the Result View is active. This tab provides key result information for the active load case.
- The **Tables** tab is used to add available tables to the report.
- The **Report Filter** tab is used to define the report settings, apply the model filters, modify the parameters of the table selected in the report, and select which table columns to display.

Graphic Views

These views provide a way to view the model, analysis results, design results, and reports. Each tab displays different options and will provide different information in the Project Manager and Find Tool. Some Graphic tabs will only appear based on objects in your model, such as the Beam diagrams.

Project Status

This panel provides a quick update on what is done, what is in-progress, and whether things are working or failing in your model or checks. Click on any item that is underlined for more information, a report, or a dialog containing quick actions.

Pipeline Status

Shows background meshing/analysis/design progress. Background processing is done on a separate thread of your processor so you may continue working while they run. The only time you need to wait for the program is when the mouse cursor changes into an hour-glass or if you wish to view the analysis or design checks that are currently in-progress. Detailed progress bars are available for background activity by clicking on the status-bar at the bottom of the screen.

Find Tool

The Find Tool provides an efficient way to view, select, and edit slabs, members, supports, loads, etc. This tool is docked on the bottom of the window by default. Use ***F7*** or the push-pin icon to auto-hide this panel. When docked, drag the side border to make the panel larger or smaller. The Find tool allows you to find, select, edit, and delete objects even if they are not visible in the active window. ***Double-click*** on an element (slab, pile, pier, etc) and the graphics window will zoom-in to show that element, if it is visible. Lists shown in the Find tool can be sorted by clicking on a column header (***click*** again to reverse the order). Select items just like any list in Windows using the ***Shift*** and ***Ctrl*** keys to select a range or to toggle individual items.

Units & Precision

Above the toolbar on the far right is the Units drop-down for selecting the way physical quantities are displayed. Change the number of decimal places or significant digits using the icon to the left of the unit selector. Go to **Home | Manage Units** to create custom unit styles or edit existing unit styles.

Data Entry: Physical Quantities

Enter values in any unit style. Enter any number or math expressions followed by a known abbreviation. Length units may be entered in "ft-in-16ths" notation as well. Entered values are converted and then redisplayed in the current 'display' units.

Window Locations

VisualFoundation will automatically remember the window and panel locations. Panels like the Project Manager or Find

Tool can be collapsed with auto-hide, or floated in a separate window. You can reset the window layouts using the **Tools | Custom Data** command to locate the *VF.DockingLayout.xml* file and delete it.

Mouse and Keyboard Commands

Selection:

- ***Click*** to select (mouse hover indicates what object will be selected)
- ***Click*** in the 'whitespace' of a view to unselect everything and access Project Settings.
- ***Ctrl+Click*** to toggle object selection without affecting other objects
- ***Shift+Click*** to select all objects of a given type.
- ***Shift+Drag*** draw a selection box (left-to right selects fully enclosed objects, right-to-left selects any partially enclosed objects)
- ***Shift+Ctrl+Click*** to select items of one type with the same Name Prefix as the item clicked on.

Zoom:

- ***Scroll Mouse Wheel*** with the pointer over the point to zoom in or out from.
- ***Double Click Mouse Wheel*** to Zoom All.
- ***Ctrl+*** (plus) and ***Ctrl-*** (minus) keys.
- ***Ctrl+Home*** for zoom all/extents
- ***Ctrl+End*** to enable the **Home | Zoom Area** command then ***Drag*** to create the Area.

Pan:

- ***Drag Mouse Wheel*** to pan.
- ***Shift+Arrow*** keys will also pan.

Rotate:

- ***Ctrl+Drag Mouse Wheel*** to rotate the view.
- ***Click*** on a face, edge or corner of the Cube in the lower-left corner of the graphics to rotate the view.
- ***Ctrl+Arrow*** keys will also rotate.

Context Menu:

- ***Right-Click*** the mouse for a short menu of relevant commands based on the view and what is selected.
- ***Shift+F10*** also display the context menu.

Hot Keys:

- ***Alt*** will expose the hot-keys in the main menu
- ***F1*** Help.
- ***F7*** Show or hide the Find Tool.
- ***F9*** Toggle Picture View
- **Esc** Cancel the Graphic drawing and enter the Draw Nothing mode.
- **Delete** the Graphic selection.
- ***Ctrl+C*** Copy graphic image to clipboard.
- ***Ctrl+V*** Generate copies, or paste graphics in Report View.

Miscellaneous:

- ***Drag*** in the Model View to sketch slabs, grade beams, walls, etc.
- ***Double Clicking*** in the Analysis Result will generate a Text Report for the object. Double-clicking on an element or node in the Find Tool will Zoom to that item.

Middle-Mouse "Button" in Windows

Depending on your system, you may need to go into Control Panel, Hardware, Mouse, and set the wheel button to behave like a "middle button click". Some mouse utility programs may override that setting or it may not be set up on some versions of Windows.

1.4 Upgrade Guide

Version 12.0 (March 2023)

Scripts & Command Line

- Command Line is a powerful new way to drive the program:
  - Definite slabs, piles, piers, grade beams, walls, etc.
  - Apply loads to regions, piers, grade beams, and walls
  - Adjust the analysis, stability, and soil settings
  - Extract plate results (displacements, shears, moments, etc.) and pile results
  - Add tables to report and export reports
- External Script files can automate common tasks and much more:
  - Generate foundations, apply loads, extract results
  - Refine the mesh until model converges
  - Perform model optimization

Other Features

- ACI 318-19 & CSA A23.3:19 concrete specifications added
- CSA S16:19 steel specification added
- Plate design diagrams
- Pier combined flexure and tension design
- Table of Contents report table

1.5 Release History

Overview

Versions

- Version 12.0 released March 2023
- Version 11.0 released September 2021
- Version 10.0 released January 2020

- Version 9.0 released September 2018
- Version 8.0 released October 2017
- Version 7.0 released August 2016
- Version 6.0 released April 2015
- Version 5.0 released January 2014
- Version 4.0 released May 2012
- Version 3.0 released June 2011
- Version 2.0 released April 2010
- Version 1.0 released April 2009

Version 12

Scripts & Command Line

- Command Line is a powerful new way to drive the program:
  - Definite slabs, piles, piers, grade beams, walls, etc.
  - Apply loads to regions, piers, grade beams, and walls
  - Adjust the analysis, stability, and soil settings
  - Extract plate results (displacements, shears, moments, etc.) and pile results
  - Add tables to report and export reports
- External Script files can automate common tasks and much more:
  - Generate foundations, apply loads, extract results
  - Refine the mesh until model converges
  - Perform model optimization

Other Features

- ACI 318-19 & CSA A23.3:19 concrete specifications added
- CSA S16:19 steel specification added
- Plate design diagrams
- Pier combined flexure and tension design
- Table of Contents report table

Version 11

Key Features

- Slab's mesh can be refined at pier and pile locations
- Punching shear reinforcement can be specified

Other Features

- Plate result diagram overhaul
- Wall loads remain constant when switching between resultant to distributed loads

- Program ensures area loads are properly modeled before starting the analysis

Fixes & Minor Changes

- Improved Help File documentation
- Reinforcement details column added to the "Slabs" report table
- Loaded Area information added to Area Uniform Loads tab in the Find Tool
- Removed concrete pile effective length factors (Kz & Ky) as concrete piles are design as short columns
- Fixed sliding stability check for models with loads causing uplift
- Corrected graphics of piles rotated around their cross-section's axis
- Corrected pier height graphics

Version 10

New Features

- Slab reinforcement can be specified (patterns can still be searched using the optimize design approach)
- The displacement, forces, and bearing pressure can be viewed at a specific location
- Improved Design Rebar Patterns dialog that retains rebar patterns
- Improved filtering of table extremes
- Report can be saved in the project or saved as a style
- Justification can be specified for report tables
- Minimum flexural reinforcement area check added
- Improved bearing pressure integration for punching shear calculations
- Improved drawing grids
- The number of shear legs can be specified for grade beams

Fixes & Minor Changes

- Improved Help File documentation
- Corrected pier rotation direction in the Foundation Design view
- Improved color bands for analysis and design results
- Subgrade modulus can be graphically displayed
- Improved pile report tables
- The pile batter feature caused the plate elements to be numerical unstable for many cases and was removed

Version 9

New Features

- Columns (terminology) replaced by Piers: with material properties, loads at top of pier, and self-weight
- Canadian CSA A23.3-14 specification added
- Design of piers (concrete with rebar details)
- Structural design of piles (NDS wood, ACI/CSA concrete, or AISC/CISC steel)

- Pile supports may be battered
- Pile sliding resistance may be specified (was previously infinite)
- Walls may have their own material
- Many performance-improvements (12% to 30x faster operations!)
- Ring and rectangular wall groups (faster single-entity definition)
- Ability to move area loads by side coordinates
- Ability to override punching perimeter
- Ability to insert a vertex along an area side
- Wall and beam loads can now be applied parallel and perpendicular
- DXF import shows bounds and an option for centering at the origin
- Graphic wire-frame in picture view mode
- Added name filters for graphics and reports
- Report View has a 'reset filters' option on toolbar
- Start screen shows thumbnail views of recent projects
- Sliding friction values shown in Results Inspector
- Side snap points added to areas, beams and walls for drawing
- Full report has optional categories: model, loads, results, design, graphics
- Filter grade beam design results graphically by limit-state

Fixes & Minor Changes

- Wall and beam linear loads are now persistent
- Regular-polygon foundation may be defined by side length or the radius
- Bearing pressures "within pile cross-section" are now excluded from checks
- Improved area load graphics with shaded tops
- Design summary report includes 'As required'
- Add report table: drop-position is recognized
- Unavailable tables shown disabled, with tip for reason they are not available
- Project Manager: drop-lists activated by clicking on text, not just arrow
- Project Manager: categories remember expanded/collapsed state
- Memory-leaks fixed (slowed program over long periods of time)
- Column-line graphics are filtered to avoid clutter
- Improved warning message formats
- Improved graphic window Print Preview
- Multiple-selection in Load Case Manager

Version 8

This version incorporates numerous technology and platform improvements from the past year's work on VisualAnalysis. These are numerous, and subtle improvements in the overall look & feel, as well as performance and stability changes.

- Engineer-specified rebar for concrete beams
- Full Unity Checks for Concrete Beams with 5 limit states
- Exports directly to VisualAnalysis 17.0+ file format

- Cleaner user-interface is less distracting to work with
- Improved meshing algorithm
- Pile displacements in pile result reports
- Snap point drawing
- Grade beam rotation angle
- DXF points imported as vertices
- Overlapping boundary subgrade handling (max, min, average, or sum)

⚠ Legacy support was removed. To open **VisualFoundation 6.0 or prior projects**, use VF 7.0 to upgrade the project-file, then it will open in VF version 8.0 or newer. If needed, contact support to obtain a download link for VF version 7.0.

Version 7

This version represents a complete "rewrite" behind the scenes, using new technologies for graphics, reporting, UI, and more. There are many fundamental changes from version 6.0, though it read the older file format and the fundamental features are very similar to prior versions. We've upgraded and changed where it made sense for consistency with other IES tools, for ease of operations, for performance, and for greater engineering control.

One of the biggest changes is **fully-automated background analysis**. Without slowing down you or your system, we take advantage of the fact that 90% or more of your CPU processing power is going to waste. After every change, VisualAnalysis will start working on analysis results and/or design checks. When you are ready to see them, just switch to the Analysis Results or Design view and they will likely already be available. If not, just wait a few seconds and they will appear, you can view progress in the status bar, if you wish, by using a preference setting.

General

- 64-bit implementation, no more out-of-memory issues
- Multiple-threaded architecture uses all processor cores
- New UI, Ribbon toolbar, consistent with other IES tools
- 3D model/viewing (ctrl+mouse wheel or click the cube)
- Preference settings (fonts, colors, sizes, options, etc.)
- History Flies (automatic daily backups of a project-file, see preferences)
- Your Logo in a Text Report (see preferences)

Modeling

- No merge/intersection necessary for footing boundaries
- Easier to edit
- Automatic split/connect meshing for crossing or overlapping walls, beams
- Control over overlapping footing thicknesses
- DXF import has insertion point so huge coordinate files can be moved.

Loading

- Implements ASCE 7 load combinations
- More Direct full-footing loading
- Full Boundary load is easier to use

Analysis

- Automated behind-the-scenes
- Much faster analysis (using all processor cores)
- More accurate results

Design

- Implements ACI 318-14 specification
- Improved grade-beam design
- Faster design checks

Reporting

- Much improved Report Viewer
- More tables and options
- Saved reports in project files
- Predefined, Customizable reports

1.6 Preferences

VisualFoundation preferences are default settings that primarily affect the behavior of new projects. These are not project-specific settings, which are found in the Project Manager. The preference settings can be adjusted through **Home | Preferences**. Some settings do not take effect until a new project is created or until the program is restarted. Use the Restore All Defaults button to restore the VisualFoundation preference settings to their original state. While most of the preference settings are self-explanatory, a few are documented below. Preference settings are saved on your machine in the IES folder: *C:\Users\<your.login>\AppData\Local\IES\Customer*.

Project, History Files

VisualFoundation can create up to 10 dated-backup copies of a project-file. These are stored in the IES\Customer\HistoryProjects folder. If you have data corruption or lost a project, you might be able to recover from one of these. You can turn this feature off (to save disk space or use your own backup mechanisms) by setting the value to zero.

Project, Next Inspector Field on Enter

By default, in the Project Manager, Modify area, when you press the Enter key it sets your input data but does not change fields. The Tab key lets you move from one field to another. You can enable the movement to the next field on the Enter key with this option.

Reports, Maximum Page Count

Reports can become quite large and create performance issues. VisualFoundation allows you to set the maximum number of pages allowed in a report before truncation occurs and a warning appears.

1.7 Support Resources

Did you Search this Help File?

Take advantage of the help and support built into the software, as described in the Program Layout section of the User's Guide. This document can be searched, and you should try different potential terms, sometimes less is more when searching (use just the unique word or words). A Table of Contents is also available.

Do Not Contact Support For:

- **Licensing/Sales.** Use www.iesweb.com or sales@iesweb.com.
- **Modeling Advice.** Determining how to model a structure is your responsibility as an engineer.
- **Model Validation.** IES cannot validate your model or your results. If you can document a software defect, contact support and we will investigate further and create fixes as necessary.
- **Engineering Theory**. IES is not in the business of educating engineers. There are textbooks referenced in this help file.

Technical Support

- **Support Email:** support@iesweb.com. Replies are usually within 2 business hours, if you don't hear anything within a business day, assume it got spam filtered or lost and follow-up. For best results, be sure to ask a question, indicate exactly which IES product & version you are using, include as much detail as is practical. If relevant, please attach a project file and/or screenshots.
- **Support Telephone:** Not Available. We have found this to be too inefficient for everybody. With email you can attach a screen shot, a project file, and we can better direct your question to the IES expert for that product or area. Phone tag takes longer than you think.
- **Business Questions:** For any licensing or sales-related questions or issues contact sales@iesweb.com.
- **Free Training Videos**: See the Table of Contents in this help file.

2 Modeling

2.1 Foundations

Concrete foundations (also know as boundaries, slabs, footings, mats, etc.) are the primary elements modeled and designed in VisualFoundation. Foundations with a wide variety of shapes and sizes can be modeled in the program. Piles, Grade Beams, and Walls are used to add stiffness to the foundations. A single concrete material is used for all slabs in the project and is defined in the **Project Settings**.

Modeling

Slabs

In the Model and Load view, circular, rectangular, polygonal, or custom slabs can be drawn using the buttons in the ribbon. Slabs are defined by the vertices at boundary points. Slabs can be drawn on grids or existing vertices and snap points can be used to create slabs. The number of snap points along each edge of the slab is specified in the in the **Project Manager | Filter** tab. Any arbitrary geometry can be constructed by drawing multiple slabs connected to or overlapping each other. Adjoining slabs are modeled as continuous, with common displacements and flexural rotations at the boundaries. Each individual slab boundary can have a different thickness and Subgrade Modulus. At locations of overlapping slabs, the thickness and soil modulus are specified using the Thickness Overlap and the Soil Modulus parameters in the **Project Manager | Modify** tab. Note: Disconnected slabs are not allowed in VisualFoundation and expansion-joints or other discontinuities in the slab cannot be modeled.

Holes

A selected slabs can be turned into a hole by setting Hole? = Yes in the **Project Manager | Modify** tab. Loads that exist within holes are not included in the analysis. If portions of Area Loads lie partially over holes, only the loading lying over slabs are considered (i.e. the loading that occurs over holes is not distributed to adjacent plate elements).

Reinforcement Details

The reinforcement parameters for each slab are defined by selecting the slab from the Model and Load view and editing the *Reinforcement Details* section in the **Project Manager | Modify** tab, as described below.

- **Bar Configuration:** Double Mat or Single Mat. Determines if X & Y reinforcement is placed in a single or double layer in the concrete slab.
    - *Double Mat Parameters*
        - **Top / Bottom Layer Outer Bars:** The direction, X or Y, of the outer reinforcing bars. Outer bars are the bars closest to the top/bottom face of the slab.
        - **Top / Bottom Cover:** The clear cover from the top/bottom face of the slab to the outer reinforcing bars.
    - *Single Mat Parameters*
        - **Top Bars:** The direction of the Top reinforcement bars. The Top bars are the bars closest to the Global +Z face of the slab.
        - **Bottom Cover:** The clear cover from the bottom face of the slab to the reinforcing bars.
- **Design Approach:** Optimize or Specify. The Optimize design approach allows VisualFoundation to search for an optimal reinforcement pattern that meets all design requirements from a pre-defined list of desired patterns. The Specify design approach allows one reinforcement pattern for each direction and layer to be specified explicitly and checked during the design process. Further information of how this affects the design process can be found below in the discussion on Design Flexure Checks.

- *Optimize Approach*
  - **Rebar Patterns:** Defines the desired rebar patterns available for the slab during the design process.
- *Specify Approach*
  - **Size:** The size of the reinforcement in the X & Y directions and for the Top and Bottom, or Single layers, depending on the Bar Configuration selected.
  - **Spacing:** The spacing of the reinforcement in the X & Y directions and for the Top and Bottom, or Single layers, depending on the Bar Configuration selected.

Loading

Foundations may be loaded with Full Area, Circular, Rectangular, Ring, and Tubular area loads. Loads can also be transferred to the slab by Piers, Grade Beams, and Walls.

Analysis

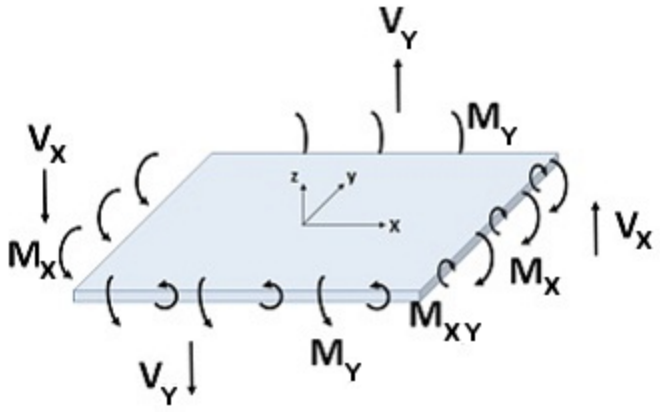
Finite element analysis is approximate and numerical. Use mesh refinement and examine the analysis results carefully, before trusting design checks.

FEA Model

Slabs in VisualFoundation are meshed into triangular FEA plate elements with displacement springs used to model the slab-soil interaction. See the Analysis page for more information on the FEA mesh density, including the available settings and options. For a better understanding of the FEA model that is built by VisualFoundation, use the **File | Export a VA Project** feature to create a VisualAnalysis project and examine the FEA model in more detail. More information on integration between VisualFoundation and both VisualAnalsyis and VAConnect can be found in the Integration section of the Help File.

Plate Results Sign Convention

1. Positive moments put the slab's top bars (Global +Z) in tension.
2. MX and VX moments and shears act on the Global X-face of the plate elements.
3. MY and VY moments and shears act on the Global Y-face of the plate elements.
4. Moments are carried by reinforcement that runs parallel to the specified directions, X & Y.



Viewing Analysis Results

The results from the finite element analysis can be displayed graphically in the Analysis Results view. The **Project**

Manager | Results tab displays the numerical results that correspond to the colored graphics. With nothing selected, the results displayed in the **Project Manager** are a summary for the selected result case, whereas if one or more individual plates are selected, the **Project Manager** shows the result range for the selected plates. The various Result Cases that were included in the finite element analysis can be selected using the Result Case drop down from the **Ribbon | Home** tab. The Result Type displayed graphically in the Analysis Result view is specified in the **Project Manager | Result Filter tab**.

Design

VisualFoundation checks and designs concrete slabs according to the following design specifications:

- ACI 318-19
- ACI 318-14
- CSA A23.3:19
- CSA A23.3-14

Flexure Checks

Slabs that experience two-way action have both bending moments (MX and MY) and twisting moments (MXY). The twisting moments may cause the maximum bending demand to not coincide with the X or Y reinforcing directions. To ensure that the slab has adequate strength in all directions, VisualFoundation uses the method by Wood and Armer as explained by MacGregor and Wight[1,2] to design the reinforcement. The following equations are used to calculate the design moments which are obtained from Wood and Armer when k=1.0. According to MacGregor and Wight,[2] k=1.0 is the best choice for a wide range of moment values. VisualFoundation takes the critical flexure section as the face of Grade Beams, Piles, Piers, and Walls. Plate nodes that fall within these objects are not checked for flexure.

$M_X+ = M_X + |M_{XY}| \geq 0$
$M_X- = M_X - |M_{XY}| \leq 0$
$M_Y+ = M_Y + |M_{XY}| \geq 0$
$M_Y- = M_Y - |M_{XY}| \leq 0$

The bending moment capacity of the slab is a function of the compressive strength (f'c), thickness, and the area of steel reinforcement provided. Reinforcing is placed in both plan directions (X & Y) and in either one or two layers. This results in up to four reinforcing quantities at every point in the slab (X, Y, Top, Bottom). VisualFoundation performs moment and steel demand calculations at the finite element model node points. When the slab's reinforcement is set to Optimize (as discussed above), the reinforcement required to meet factored demand is selected by the program based on the list of available rebar patterns. The rebar pattern with the smallest area (As) is selected from this list that meets the required flexural and minimum steel demands. If the slab's reinforcement is set to Specify, the reinforcement pattern defined for the direction and layer in question will be used when performing the design checks.

One Way Shear Checks

The critical section for one-way shear (i.e. beam shear) is taken at the face of Piles, Piers, Walls, and Grade Beams. If, however, a Pile or Pier introduces compression in the slab, the critical section is taken at "d" from the face of the object.

Plate shear checks are reported graphically from the Foundation Design view by selecting *Shear Unity* from the **Project Manager | Design Filter** under the *Slab Details* category (red plate colors indicate a failing unity value). Shear checks can also be reported as numerical results using the programs's Test Reports.

Punching Shear Checks

Two-way (i.e. punching shear) checks are made at locations where a concentrated load is introduced into the slab, such

as at Piles, Piers, and Walls. The design check for punching shear involves calculating the punching perimeter (i.e. Critical Section) around the object, which may be truncated by the slab boundary. The nominal strength of the punching perimeter must exceed the net load acting inside the perimeter.

If a Grade Beam crosses an object's punching perimeter, punching shear checks are not performed ("-N.A.-" will appear in the punching shear report). In this situation, it is assumed that the Grade Beam will be designed to carry the required shear demands. Punching shear checks are not performed for rectangular/circular loads or for beams.

The punching perimeter can be shown graphically on both the Model and Load view and the Foundation Design view. When punching reinforcement is present (see below), the Outer Critical Section is displayed graphically for the object with reinforcement.

Punching Shear Groups

When Piles or Piers contact other surrounding punching perimeters, these are combined into a larger punching area. The same capacity checks are made on the larger combined perimeters. While these combined areas are shown graphically in the Model and Load view, they cannot be directly overridden or manipulated. Note: When the Override Perimeter Offset parameter (see below) is used for an individual object that is part of the punching shear group, it has an effect on the overall punching perimeter for the group as a whole.

Punching Reinforcement

Two-way shear reinforcement can be defined for individual Pier or Pile members in the form of headed shear studs or shear stirrups. When punching reinforcement is provided, a design check is performed on both the Inner and Outer Critical Section. The slab's capacity at the Inner Critical Section is determined as the sum of the concrete and steel reinforcement capacity. The slab's capacity at the Outer Critical Section is determined from the concrete capacity alone.

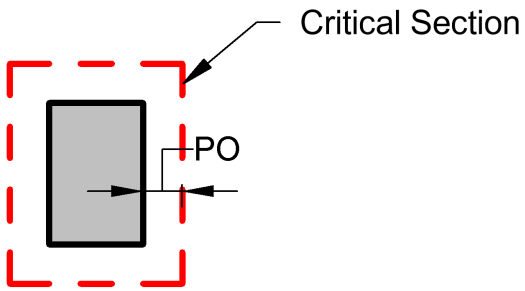
Note: The punching shear reinforcement is assumed to extend away from the object in the Global +X, +Y, -X & -Y directions, regardless of the object's shape or orientation. If a foundation edge (either an outside edge or hole edge) is located within the Reinforcement Offset distance (see below), the reinforcement is assumed to extend to the edge of the foundation in that direction.

Punching Design Parameters

Below are the design parameters available for an individual Pile or Pier object. These parameters can be found from the **Foundation Design | Modify** tab when a Pile or Pier is selected.

Piers & Piles Without Reinforcement

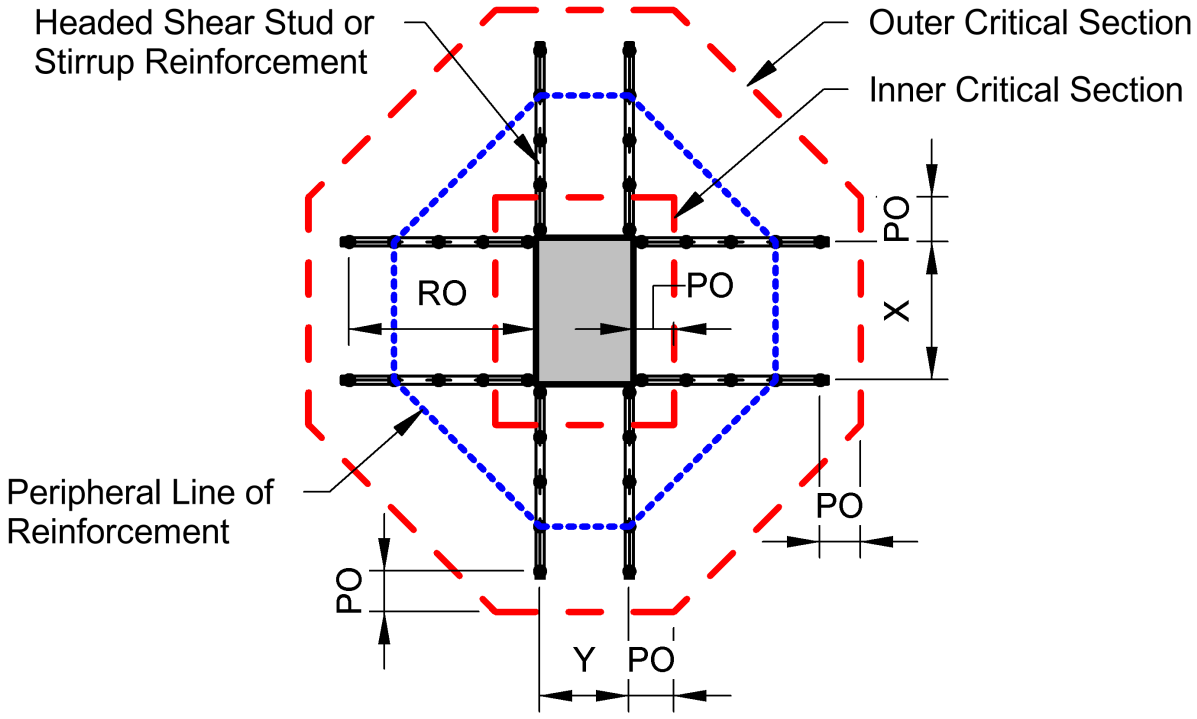
Override Punching Offset (PO)?	Do you want to override the punching perimeter offset? - *Choosing 'No' results in using the slab reinforcement depth (d/2) for the offset. Choosing 'Yes' allows you to enter your own value to be used as the punching perimeter offset.*



Piers & Piles With Reinforcement

Override Punching	Do you want to override the punching perimeter offset? - *Choosing 'No' results in using the*

Offset (PO)?	*slab reinforcement depth (d/2) for the offset. Choosing 'Yes' allows you to enter your own value to be used as the punching perimeter offset.*
Are Headed Studs?	Is the reinforcement headed shear studs?
Fyt	The yield stress of the reinforcement.
Av total	The total area of shear reinforcement around a peripheral line of reinforcement.
Spacing	The spacing of peripheral lines of shear reinforcement.
X Reinf. Group Width (X)	The width of the stud/stirrup reinforcement group oriented in the Global X-Direction. Enter 0.0 for a single stud rail or stirrup leg line.
Y Reinf. Group Width (Y)	The width of the stud/stirrup reinforcement group oriented in the Global Y-Direction. Enter 0.0 for a single stud rail or stirrup leg line.
Reinforcement Offset (RO)	The distance from the object face that the shear reinforcement extends outward.



Rebar Design Regions

The software can detail bar patterns by plate element, by column lines or by slab boundary. Use the **Project Manager | Design Filter** tab from the Foundation Design view to display the different results. Note: The *By Column Line* display is not available if the *Design Approach* is set to *Specify*.

Reporting Results

Reports can include both text-based and graphical information. Graphical information from the Analysis Results or Foundation Design views or from the plate diagrams can be inserted into a report using the **Copy** and **Paste** commands.

Plate Diagrams

Plate result diagrams for moment, shear, displacement, and bearing are available in the Analysis Results view and plate design diagrams for shear unity and area of steel required are available in the Foundation Design view. Left-click and drag a line across the slab to view the diagrams at the line's location or right-click and select Show Plate Result/Design Diagram from the context menu to launch the respective diagram dialog box. The slice direction and location can be adjusted within the dialog along with the number of plots displayed and the plot type. Note: The slice direction and location along with the plot settings will persist within one session of VisualFoundation. This allows the dialog to be closed to select a different result case or to modify the model without losing the adjustments that were made in the dialog. To include graphs in the report, use the copy plots button to copy the plots to the clip board which can then be pasted into the report.

Text Reports

Analysis and design results can also be viewed in text form using the Report View tab. Once in the Report View, a list of available deign tables is shown on the **Project Manager | Tables** tab and can be added to the text report by double-clicking an individual table or dragging and dropping a table into the report. Plates can also be reported on an individual basis by selecting one or more plates from the Analysis Results or Foundation Design view, and using the right-click context menu to *Report Selected*. Note: The reinforcement settings are included in the *Slabs* table located under the *Structure Tables* category.

References

1. Wood, Randal H. and Armer, G. S. T., "The Theory of the Strip Method For Design of Slabs," Proceedings, Institution of Civil Engineers, London, Vol. 41, October 1968, pp. 285-313.
2. MacGregor, James G., and Wight, James K., "Reinforced Concrete: Mechanics and Design," Pearson, 2012.

2.2 Pile Supports

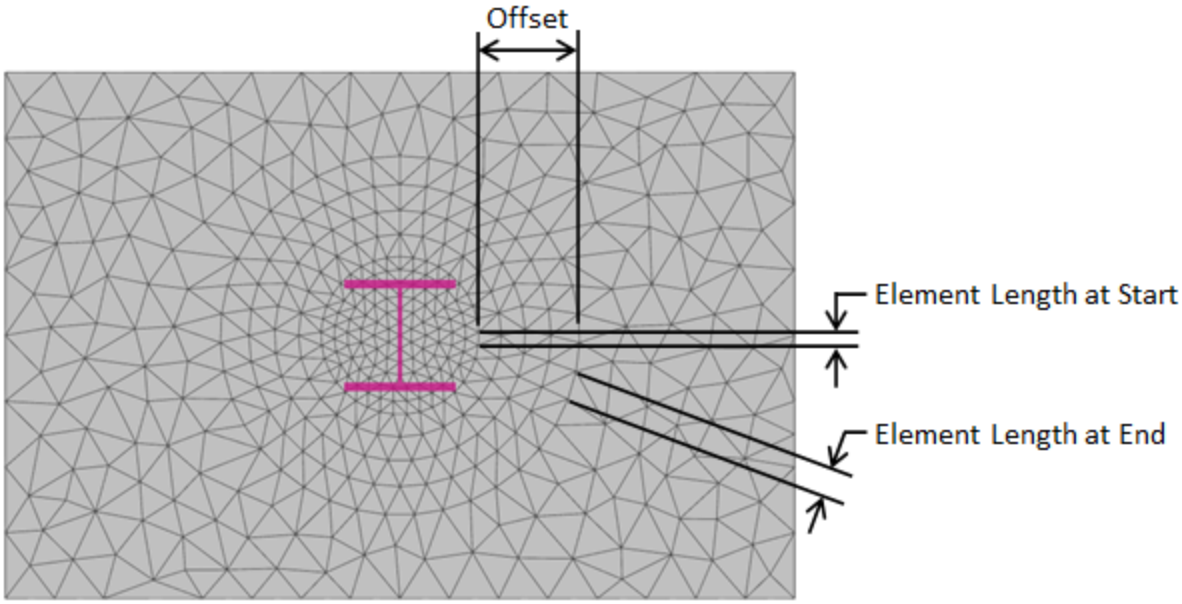
Piles in VisualFoundation are used to support the foundation at a specified location in the model.

Modeling

Piles are defined by a location, cross-section, material, and length. A Pile's cross-section can be defined by standard parametric shapes or a shape can be chosen from the database. Note: In the Finite Element Analysis (FEA) model, a pile supports a single node in the slab's mesh; therefore, care should be taken when specifying piles with large cross-sections.

Mesh Refinement

The finite element analysis (FEA) method is approximate and the accuracy of the FEA solution typically increases as the mesh becomes finer. Generally, a finer mesh should be used in regions where there are large changes in stresses in the plates (such as at piles where the support can create stress concentrations). Select a pile(s) and set Refine? = True in the **Project Manager | Modify** tab to refine the mesh in the regions of the chosen pile(s). The mesh refinement parameters for piles are defined in the image below.



Pile Support Mesh Refinement

Sliding Stability

For the Sliding Stability checks, piles can be set in the **Project Settings** to have either an infinite sliding resistance (i.e. the total sliding resistance of the foundation is infinite) or a total resistance of the piles can be manually specified. When a total pile resistance is entered, it will be added to the frictional resistance and the passive resistance when calculating the factor of safety for sliding.

Loading

Piles cannot be directly loaded in VisualFoundation. A pier can be defined at the same location as the pile (i.e. connected to the same node in the finite element model) and load can be applied to the pier.

Analysis

VisualFoundation calculates the default "Rigid" stiffness for the pile as AE/L which can be overridden to specify a custom stiffness. Piles are treated as axial-only spring elements during analysis and the action can be set to two-way, compression only, or tension only. Note: There are no soil-structure considerations for piles in VisualFoundation.

Design

In VisualFoundation, steel, wood, and concrete piles can be designed according to the following design specifications:

- ACI 318-19
- ACI 318-14
- CSA A23.3:19
- CSA A23.3-14
- AISC 360-16 ASD & LRFD
- AISC 360-10 ASD & LRFD
- AISC 360-05 ASD & LRFD
- CSA S16:19

- CSA S16-14
- NDS 2018 ASD & LRFD

Geotechnical Analysis & Design

In VisualFoundation, the geotechnical capacity is manually specified for both tension (uplift) and/or compression. These parameters represent the force at which the soil supporting the pile fail (values are typically obtained from a geotechnical engineer or report). A unity check with either service or strength-level loads is performed to compare the actual demand against the specified capacity.

Structural Design

The pile design parameters (bracing, reinforcement for concrete, C-factors for wood, etc.) are specified in the **Project Manager | Modify** tab when the **Foundation Design** view is active. In VisualFoundation, piles only carry axial forces and the axial demand is checked against the applicable limit state for the appropriate material and design specification chosen. Note: VisualFoundation does not consider the effects of slenderness for concrete piles. Therefore, concrete piles are designed as short columns and moment magnifiers are not used.

Punching Shear

In VisualFoundaiton, Punching Shear design checks are performed at pile locations and the program can automatically determine the punching shear perimeter (critical section). An individual pile can also define punching shear reinforcement in the form of headed shear studs or shear stirrups located in the slab. Additional information regarding punching shear design checks and punching shear reinforcement can be found in the Foundations - Design section.

Reports

Report tables, which include intermediate details are available for each check and/or limit state.

2.3 Column Piers

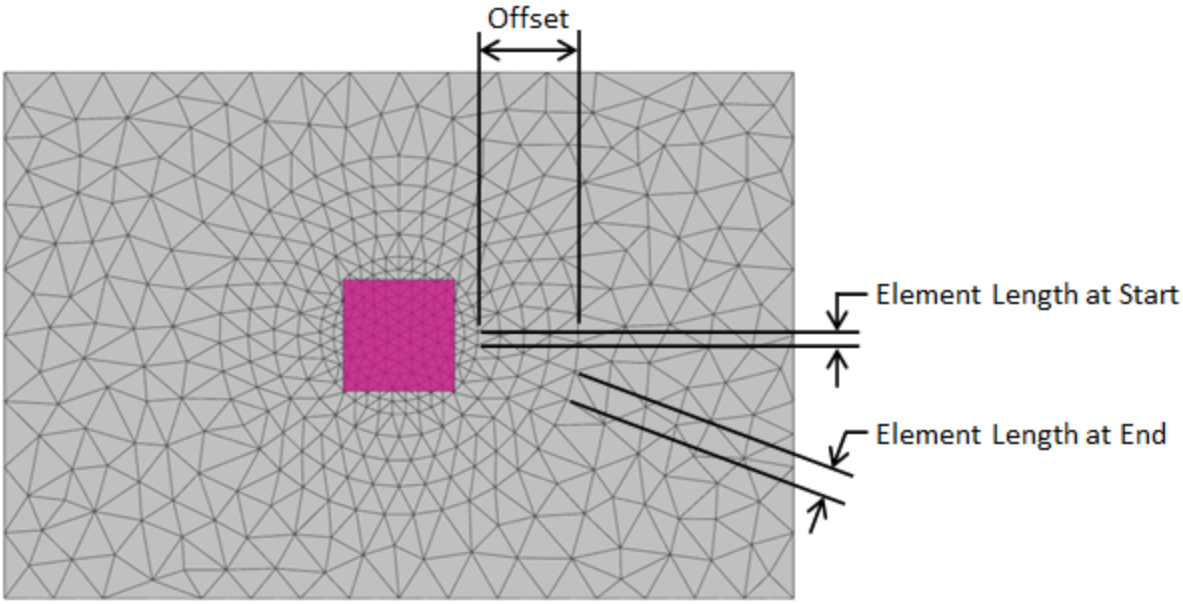
Column-Piers elements in VisualFoundation are used to apply vertical loads, sliding forces, and moments to foundations at a specified location.

Modeling

Piers are defined by their location, size, shape, height, and material properties. Four shapes are available: Square, Rectangular, Circular and Octagon. The radius of an octagon pier is based on the octagon being inscribed inside a circle with the specified radius. Note: In the Finite Element Analysis (FEA) model, a pier loads a single node in the slab's mesh; therefore, care should be taken when specifying column piers with large cross-sections.

Mesh Refinement

The FEA method is approximate and the accuracy of the FEA solution typically increases as the mesh becomes finer. Generally, a finer mesh should be used in regions where there are large changes in stresses in the plates (such as at piers where concentrated loads can create stress concentrations). Select a pier(s) and set Refine? = True in the **Project Manager | Modify** tab to refine the mesh in the regions of the chosen pier(s). The mesh refinement parameters for piers are defined in the image below.



Column Pier Mesh Refinement

Loading

Vertical forces (compression and tension), in addition to sliding forces and moments in two directions can be applied to piers. The pier's self weight can be included by checking the option for the selected pier in the **Project Manager | Modify** tab. The sliding forces are assumed to act at the top of the pier and will generate moments in the slab. Sliding forces are also used for the sliding checks on the foundation.

Analysis

During the finite element analysis each pier is modeled as a member element cantilevered from the slab mesh. Loads are applied to the top of the pier. Piers are assumed to be unsupported over their entire length.

Design

Only concrete piers can be designed in VisualFoundation and no other material can be specified for piers. VisualFoundation checks and designs concrete Piers according to the following design specifications:

- ACI 318-19
- ACI 318-14
- CSA A23.3:19
- CSA A23.3-14

Pier reinforcement (longitudinal and transverse) is specified in the **Foundation Design** view for the selected pier(s). The following limit states are checked:

- Maximum Steel
- Minimum Steel
- Bar Fit
- Flexural Shear (strong and weak axis)
- Tie Spacing
- Axial Compression

- Axial Tension
- Strong Bending + Weak Bending + Axial Interaction

Note: VisualFoundation does not design a piers when its height is set to zero (zero-height piers can be used to apply loads at the top of the slab). To preserve the legacy load definition, "columns" coming from VisualFoundation Version 8.0 and earlier are given a zero height automatically. Piers with an octagonal cross-section are cannot be designed in VisualFoundation.

Punching Shear

In VisualFoundaiton, Punching Shear design checks are performed at pier locations and the program automatically determines the punching shear perimeter (critical section). An individual pier can also define punching shear reinforcement in the form of headed shear studs or shear stirrups located in the slab. Additional information regarding punching shear design checks and punching shear reinforcement can be found in the Foundations - Design section.

Report

Reports are available to show the results for each of the applicable limit states as well as interaction diagrams for piers in VisualFoundation.

2.4 Grade Beams

Modeling

Grade beams are created in VisualFoundation by setting the drawing mode to "Draw Grade Beams" from the **Ribbon | Foundation** tab and sketching them on top of an existing slab in the model. Grade beams are used to distribute applied loads to the concrete boundary and add stiffness to the model. Grade Beams used the concrete compressive strength (f'c) defined for the project, which can be found in the **Project Manager | Modify** tab when nothing is selected. Grade beams that adjoin one another, regardless of orientation, are modeled such that load transfer (shear and moment) occurs between the members (i.e. there are no end release options for grade beam elements).

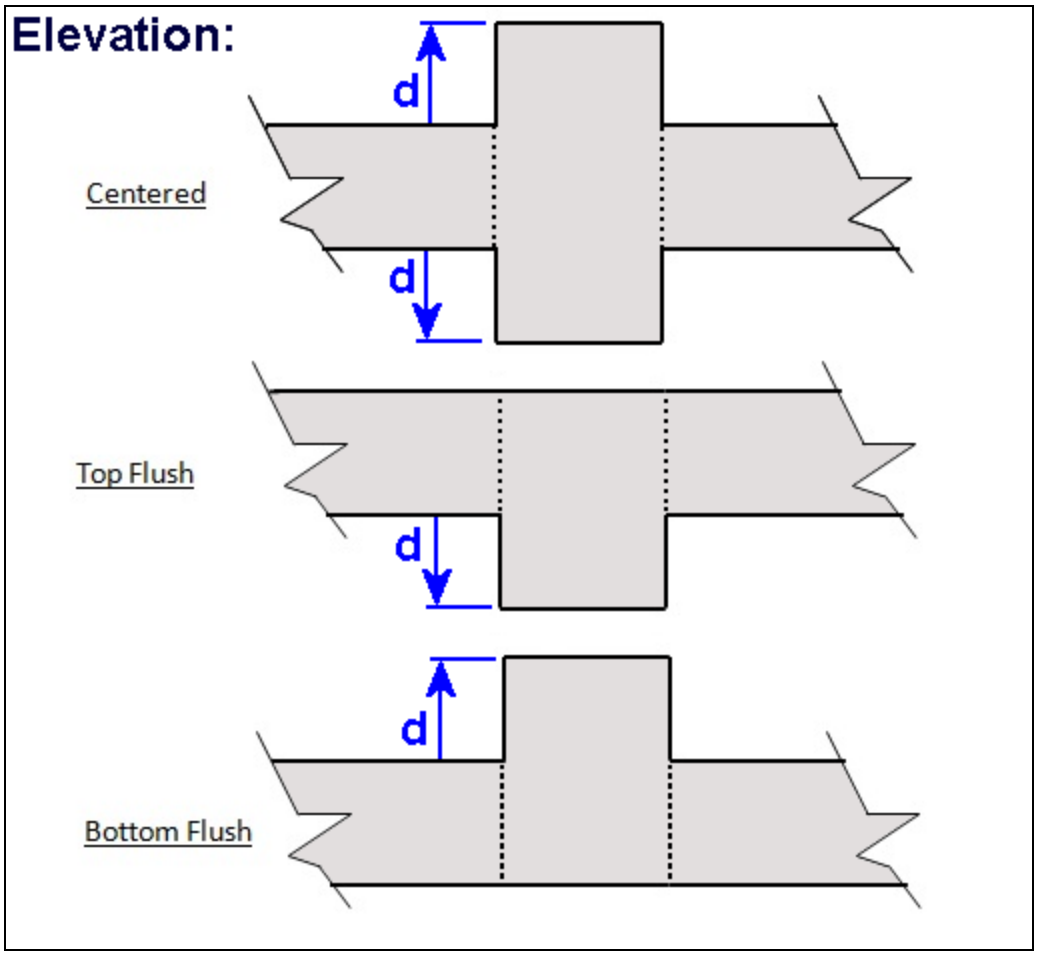
Grade Beam Subgrade

In VisualFoundation, the value of the subgrade modulus beneath a grade beam can be increased or decreased from the value used for the slab containing the grade beam. Select one or more grade beams from the **Model and Load** view, set the *Change Subgrade* parameter to "Yes" from the **Project Manager | Modify** tab, and enter a value for the Subgrade Increment. Only the nodes connected to the grade beam will be affected in the FEA model (i.e. only a single line of springs will be modified).

Beam Properties

Grade beams are rectangular in shape and have a specified Beam Width and Projected Depth. All beams have a Vertical Offset Type and Length (defined by the start and end locations).

⚠ The **Centered** vertical offset option uses **twice the specified depth**, as shown below.



Loading

Vertical forces, in addition to sliding forces and moments in two directions can be applied to grade beams. The self weight of the grade beam can be included by checking the option for the selected beam in the **Project Manager | Modify** tab. The sliding forces are assumed to act at the top of the slab and will generate moments in the slab. Sliding forces are also used for the sliding checks on the foundation. Grade beam loads can be entered as distributed evenly along beam length or as the resultant total. Loads can be entered in the global coordinate system or in the grade beam's local coordinate system (defined as parallel or perpendicular to the beam).

Analysis

In the finite element model, grade beams are modeled using member elements in the same plane as the plate elements used to model the slab. The stiffness of the grade beams are adjusted based on the Vertical Offset setting (i.e. the parallel axis theorem is used to modify the stiffness).

The results from the finite element analysis can be displayed graphically in the **Analysis Results** view. The **Project Manager | Results** tab displays the numerical results that correspond to the colored graphics. With nothing selected, the results displayed in the Project Manager are a summary for the selected result case, whereas if a single grade beam is selected, the Project Manager shows the result range for the selected grade beam. The various Result Cases that were included in the finite element analysis can be selected using the *Result Case* drop down from the **Ribbon | Home** tab. Furthermore, moment, shear, and displacement diagrams for each result case can be viewed using the Grade Beam Graph tab. Analysis results can also be reported in tabular form using the Text Reports.

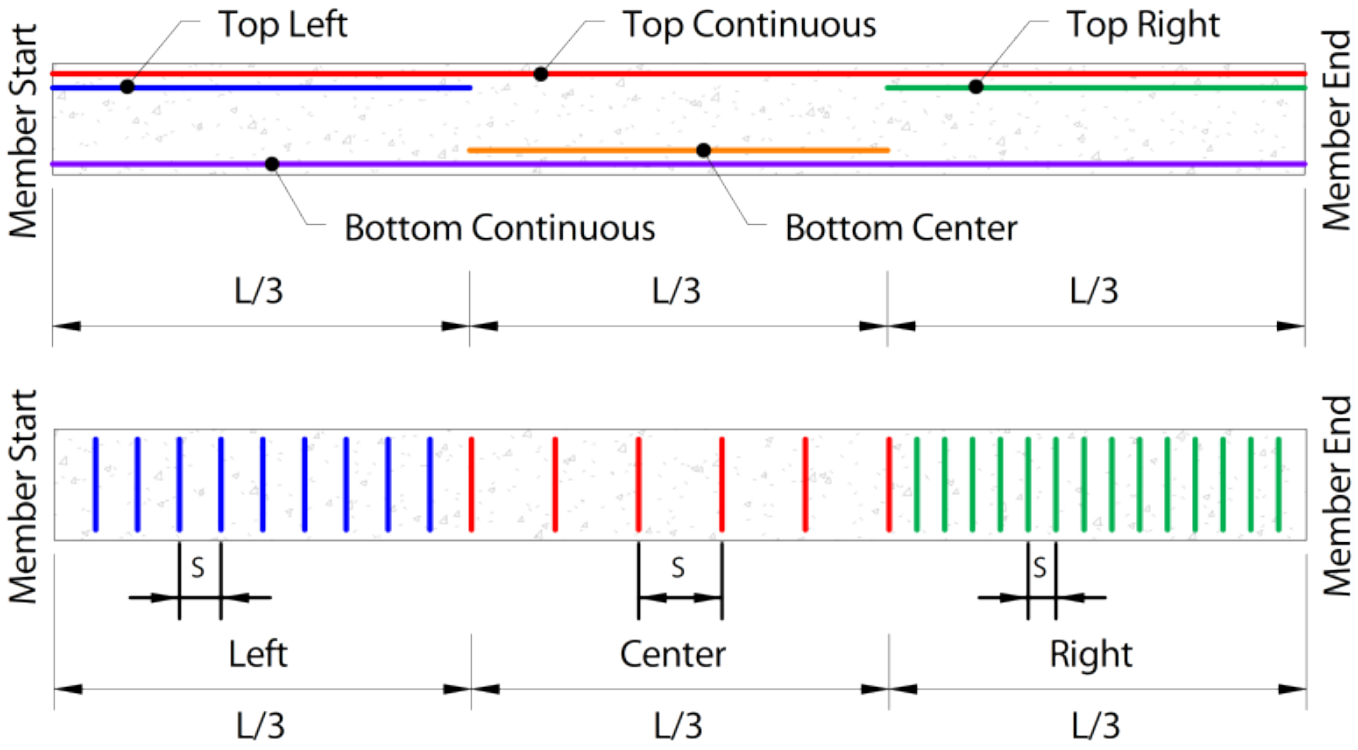
Design

VisualFoundation checks and designs concrete beam members according to specifications listed below.

- ACI 318-19
- ACI 318-14
- CSA A23.3:19
- CSA A23.3-14

Reinforcement Regions

Longitudinal and transverse reinforcing vary along the length of the member, as shown below (respectively).



Beam Assumptions and Limitations

- Axial forces are not considered.
- Rebar is calculated for each 1/3 span. Rebar is assumed to be terminated at the 1/3rd points of the beam, cutoff locations are not determined, and development length issues are not considered in the software.
- No "diagram" is produced to graphically display the bar locations or stirrup arrangements. The specified reinforcement for each beam can be displayed using Design Tables in the Text Reports.
- Deep beams cannot be designed in VisualFoundation and a warning will be provided if a beam has a clear-span length less than 4h.
- Grade beams are designed as rectangles, not T-Beams.

Design Parameters

When designing beams, several parameters must first be defined. These parameters can be set by selecting a beam in the Foundation Design view and using the **Project Manager | Modify** tab.

Beam Details

Specification - The Design Specification used to design the grade beam.

Disable Checks? - Causes selected design group to be omitted from design checks.

High Seismic? - (ACI Only, Use Reduced φ Factors for Members Resisting Earthquake Effects) Enabling this parameter lowers the φ factors as indicated by ACI 318 Section 21.2.4 for members that are designed to resist earthquake effects and are part of a structure that relies on special moment resisting frames or special structural walls to resist earthquake effects. VisualFoundation relies solely on this parameter in determining whether or not to use reduced φ factors (it does not attempt to calculate whether the shear capacity is greater than the shear corresponding to the development of the nominal flexural strength of the member). Only shear φ factors are influenced by this parameter for design according to the ACI specification.

Overstrength? - Causes the member to be designed using overstrength load combinations.

Check Level - Determines the level of detail reported from design checks. Options are: *To Failure* (Fastest), *Each Limit State*, and *All* (Slowest, but provides the most information).

Start/End Column Widths - Widths of supporting columns at start and end of grade beam. These widths are used for determining where critical moment, shear, and torsion are at the ends of the beam. The critical demands are taken at the face of the column since member's effective depth significantly increases once the column is reached. These ends correspond to the member's local axes, where the local x-axis goes from the start-node to the end-node. Note that the critical section for shear can be taken "@ d" from the face of the support using the "Shear "@ d" from start/end" parameter (below).

Shear "@ d" from start/end - When enabled, the shear value calculated "@ d from the face" of the support is used for the shear and torsion checks when the check location is between the face of the support and d. Note that when using the CSA design specification, "dv" is used instead of "d".

Reinforcement Details

Use Metric Bars - Should metric reinforcement be used instead of imperial bar sizes?

Longitudinal F_y: Specified yield strength of the longitudinal reinforcement in the grade beam.

Top, Bottom, Side (Torsional) Reinforcing

Top Reinforcement - The size and quantity of Top reinforcement throughout the beam.

Bottom Reinforcement - The size and quantity of Bottom reinforcement throughout the beam.

Side Reinforcement - The size and quantity of Side reinforcement throughout the beam per each side. Note side bars are not used to resist flexure.

Shear Reinforcement

F_y - Specified yield strength of the stirrups in the grade beam.

Size - The size of transverse reinforcement.

Number of Legs - The number of legs used for shear reinforcement.

Are Stirrups Closed? - Are closed stirrups used throughout the section?

Spacing - The center-to-center spacing of the stirrups. A different stirrup spacing can be specified for each 1/3 of the beam length. A spacing of 0 means no stirrups are provided.

Beam Cover

Top Cover - Concrete clear cover at the top of the section. Calculated as the distance from the top of the stirrup to the top of the section.

Bottom Cover - Concrete clear cover at the bottom of the section. Calculated as the distance from the bottom of the stirrup to the bottom of the section.

Side Cover - Concrete clear cover at the side of the section. Calculated as the distance from the

side of the section to the stirrup.

Displaying Design Results Graphically

Grade beam results can be viewed graphically in the Foundation Design view. By default, the controlling unity value is displayed for grade beams. The unity results for a specific design check (such as the *Shear Check, Flexure Check, Torsion Check*, etc.) can be viewed by changing the *Design Information* parameter under the Grade Beam Details category of the **Project Manager | Design Filter** tab.

Reporting Beam Design Results

Design results can also be viewed in text form using the Text Reports tab. Once on the Text Report tab, a list of available deign tables is shown on the **Project Manager | Tables** tab and can be added to the text report by double-clicking an individual table or dragging and dropping a table into the report.

2.5 Walls

Modeling Walls

Walls in VisualFoundation are used to distribute applied loads and add stiffness to the foundation. Walls are created by drawing them on a slab in the **Model and Load** view. Walls have specified thickness, material, length (defined by the start and end locations), and height.

Loading Walls

Vertical forces (compression and tension), in addition to sliding forces and moments in two directions can be applied to walls. The wall's self weight can be included by checking the option for the selected wall in the **Project Manager | Modify** tab. The sliding forces are assumed to act at the top of the wall and will generate moments in the slab. Sliding forces are also used for the sliding checks on the foundation. Wall loads can be entered as distributed evenly along wall length or as the resultant total. Loads can be entered in the global coordinate system or in the wall's local coordinate system (defined as parallel or perpendicular to the wall).

Analysis of Walls

Walls are modeled as a beam-element in the Finite Element Analysis model. While stiffness is modeled, no stresses are reported.

Wall Design

Walls are not designed in VisualFoundation, they are only used to define loads and add stiffness to the foundation.

2.6 Load Combinations

VisualFoundation uses both service level and strength level load combinations to analyze and design foundations as outlined in the table below. Several sets of both ASD and LRFD building code load combinations are built into VisualFoundation. Custom load combination sets may also be created.

Load Combination	Stability Checks	Soil Checks	Concrete Checks	Pile Checks

Set

Service Level	Yes	Yes	No	Yes
Strength Level	Yes	Yes	Yes	Yes

Custom Load Combinations

Use the **Create Factored Combination** button located in the **Load Case Manager** to create any custom combination needed. Custom factored load combinations can be imported from the clipboard using the **Import From Clipboard** button in the Load Combinations tab in the **Load Case Manager**. Text must be tab delimited and copied to the clipboard in the following format:

{ComboName} {Factor} {ServiceCaseName} {Factor} {ServiceCaseName2} ...

For example:

ComboName	1.2	D	1.6	L	0.5	Lr
MyCombo	0.9	D	1.3	W		

Seismic Criteria

In order to correctly generate load combinations that contain seismic loads, several additional parameters are required. Please refer to ASCE 7, Section 12.4, for how these parameters are used in generating load combinations. Note: These parameters are only used to generate load combinations. For example, ASCE 7 says that Seismic Category A does not require the combined orthogonal direction combinations (e.g., X + 30%Y), but Category D does.

2.7 Analysis

VisualFoundation uses finite element analysis (FEA) to determine the displacements, shears, and moments in the foundation system. Exporting the model as a VisualAnalysis project (File | Export a VA Project) will show what is happening behind the scenes in VisualFoundation. While VisualFoundation attempts to handle many details of the FEA method, some knowledge of FEA is still required to use the program successfully.

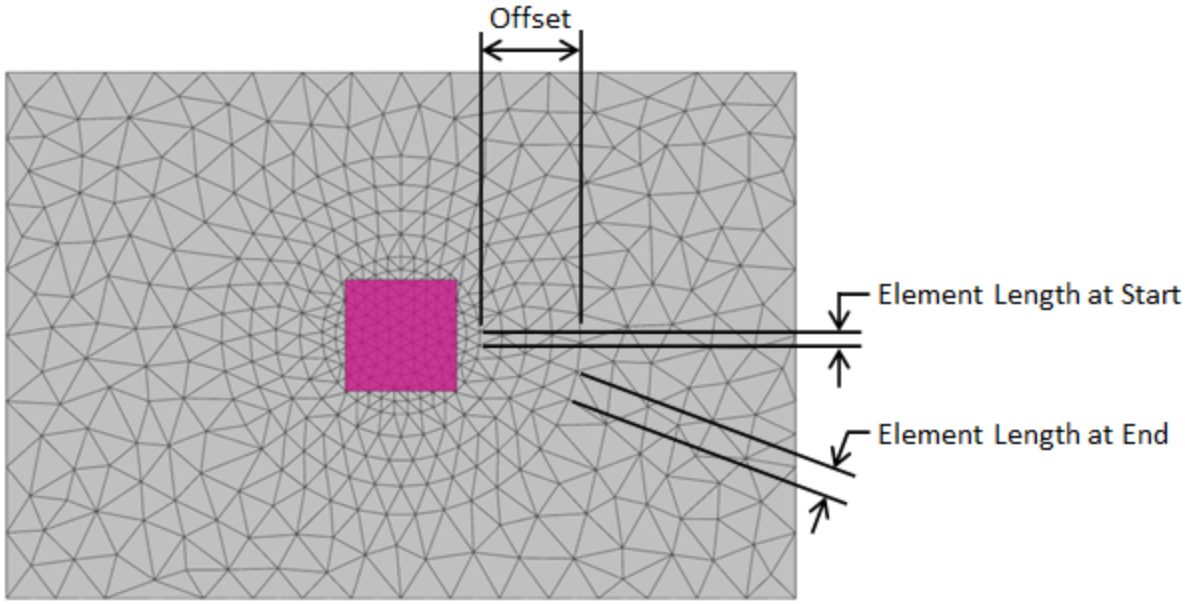
Mesh Settings

To control the number of plate elements, pick between course, medium, fine, or specify a desired number directly in the **Project Settings**. The actual number of meshed plates generated in the model is shown under Meshed Plates. Turn on Meshed Plates in the **Project Manager | Filter** tab to view the meshed plates in the model. In VisualFoundation, the advanced finite element meshing options can be modified if needed. Note: The mesh settings at Column Piers and Pile Supports can by set by selecting the model object(s) and adjusting Mesh Refinement settings in the **Project Manager | Modify** tab.

Mesh Refinement

It is the user's responsibility to verify and validate the results obtained from VisualFoundation. The finite element method is approximate, and the accuracy of the solution depends on how fine the mesh is in the model (generally, a finer mesh produces more accurate results). A finite element mesh refers to the multiple plates that are used to model a single component (such as a slab). Mesh Refinement is the process of reanalyzing the model with successively finer and finer meshes and comparing the results between these different meshes. As the mesh is refined, the change in the solution becomes smaller and an asymptotic behavior of the solution starts to emerge as shown in the figure under Mesh Refinement Procedure below. Eventually, the changes to the solution will be small enough that engineering judgment can be used to determine that the model has converged.

In VisualFoundation, mesh refinement is accomplished by reducing the Element Count in the **Project Manager | Modify | Project Settings**. Furthermore, as shown in the image below, the mesh can be refined at Column Piers and Pile Supports by selecting the model object(s) and adjusting Mesh Refinement settings in the **Project Manager | Modify** tab. In general, a finer mesh should be used in areas where there are large changes in stresses in the plates (e.g. pier locations that receive concentrated loads and pile locations where the support at a single node can create stress concentrations).



Column Pier & Pile Support Mesh Refinement

Mesh Refinement Procedure

1. Model the slab and VisualFoundation will automatically generate the plate elements based on the Mesh Refinement settings.
2. Let the analysis run and record the results.
3. Increase the number of Meshed Plates for the slab and/or refine the mesh at Column Piers and Piles Supports as discussed above.
4. Let the analysis run and record the results.
5. Compare the results from Step 4 with the results from Step 2. If the difference in the analysis results is small and acceptable (using engineering judgment), the mesh refinement process is complete. If the difference in the analysis results is large and unacceptable (using engineering judgment), start back at Step 3.

Note: Model results such as displacement, moment, shear, etc., which are represented as Phi in the graph below, will coverage at different rates. Therefore, it is important to ensure that the model has converged for the result of interests.

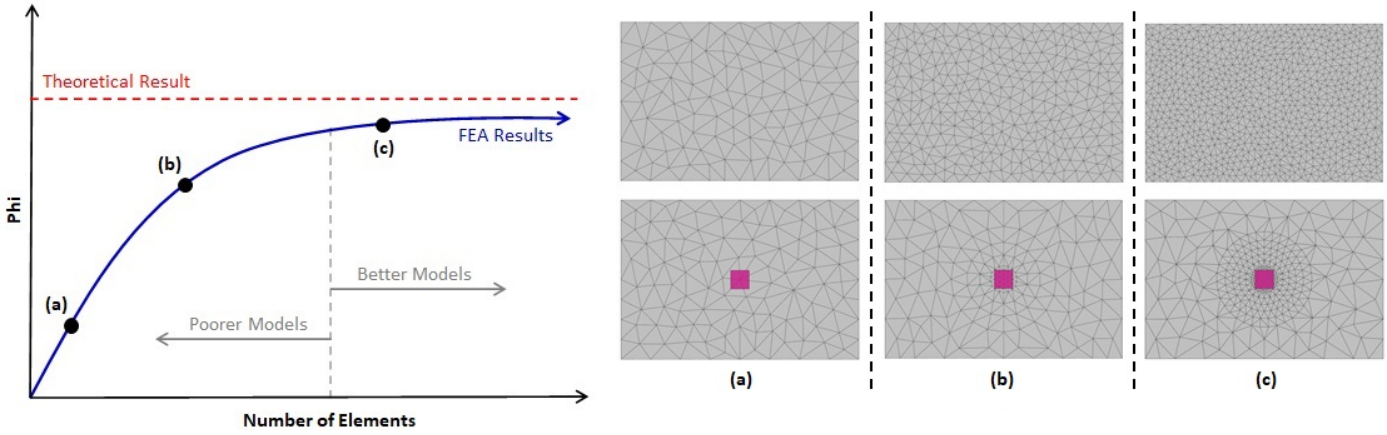


Plate Bending

The bending part of the FEA plate element is based on the triangle formulation originally presented by Xu et. al.[1] in 1992. This element accounts for transverse shear effects present in structures that might contain areas with thick plates, such as footings or thick floor slabs.

Statics Checks

A Statics Check is performed for each load case analyzed in VisualFoundation. The total applied loads in each global direction is calculated and compared to the sum of all support reactions in the corresponding global directions. The applied loads are based on the deformed shape of the structure while the reactions are based on the structure's undeformed shape. If the loads and reaction are equal and opposite in magnitude, then the structure is in equilibrium. An imbalance indicates that the deflections are large enough to generate inaccurate results which might indicate that there is a modeling problem. VisualFoundation provides a warning if a significant imbalance is detected. If a warning is received, carefully review the model to ensure it is set up correctly and verify the results. The Statics Check is displayed on the **Project Manager | Results** tab or the Statics Check Information table can be added to the report.

What to Look For?

Check to see if displacements and stress-levels in the model are reasonable and expected. A large force imbalance, either in percentage or in absolute value is a problem that cannot be ignored. Errors are usually caused by large displacements or rotations which can result from global or localized problems.

References

1. Xu, Zhongnian, "A thick-thin triangular plate element" *International Journal for Numerical Methods in Engineering*, Vol. 33, 1992, pp. 963-973.

2.8 Stability

General

Before the internal behavior of a footing can be analyzed, an engineer must make sure the footing will not lift off, tip over, or slide on the soil. This is done by providing a minimum factor of safety against uplift, overturning, and sliding. VisualFoundation allows these minimum factors to be set in the **Project Settings** (visible in the **Project Manager | Modify** tab when nothing is selected) under the **Stability Calculations** section. Select whether stability is checked with the service-level or the strength-level loads and specify the factor of safety (F.S.) for overturning, uplift, and sliding.

When piles are present, VisualFoundation assumes the footing has a sufficient factor of safety for all stability conditions (note: the infinite pile resistance for sliding can be disabled in the **Project Manager | Modify** tab and a value for the pile lateral resistance can be specified). By anchoring a footing with piles, stability requirements are satisfied by ensuring pile reactions are less than each pile's capacity.

Uplift

Uplift forces (F_{uplift}) must be less than the forces resisting uplift (F_{weight}). In VisualFoundation, the forces that resist uplift are the self-weight of the footing and its components (i.e. the slab, grade beams, walls, etc). When a footing is supported by piles, the F_{weight} is assumed to be infinite, and the uplift stability is not checked. It may be appropriate to check the uplift capacity including the pull-out capacity for the piles outside of the program. The factor of safety (F.S.) for uplift stability is defined by the following equation:

F.S. uplift = F_{weight} / F_{uplift}

Overturning

The lateral forces that a structure resists from wind, seismic loads, etc. tend to overturn the structure which in turn tends to overturn footings. Uneven pressures and forces acting on the top of the footing can also contribute to overturning. Checking the overturning stability involves calculating the overturning moment ($M_{overturning}$) and the resisting moment ($M_{resisting}$) of the forces acting on the footing. The critical locations used to calculate $M_{overturning}$ and $M_{resisting}$ are the tipping points which are taken as the corners of the footing. The axes which moments are summed are the axes parallel to the edges of the footing at the tipping points. The factor of safety (F.S.) for overturning stability is defined by the following equation:

F.S. overturning = $M_{resisting}$ / $M_{overturning}$

VisualFoundation allows the effects of seismic overturning to be reduced as allowed by ASCE 7-16 section 12.13.4. This reduction is accounted for by setting the Reduce Seismic Ovrtrn.? parameter to Yes, in the **Project Settings** (visible in the **Project Manager | Modify** tab when nothing is selected) under the **Stability Calculations** section. Enabling this parameters reduces the overturning moments generated by seismic loads by 25% before they are combined with the overturning moments from other load sources. These reduced loads are only used when calculating the overturning factor of safety and full seismic loads are used for all other analysis and design purposes. The 25% reduction assumes the structure was analyzed using the Equivalent Lateral Force procedure (ASCE 7-16 section 12.8). While ASCE allows a 10% reduction for structures analyzed using the Modal Analysis procedure, this lesser reduction is currently not supported by VisualFoundation. Note: ASCE does not allow this reduction for inverted pendulum or cantilevered column type structures.

Sliding

The lateral forces that a structure resists from wind, seismic loads, etc. can cause the structure's foundation to slide. Checking the sliding stability involves calculating the sliding forces ($F_{sliding}$) and the sliding resistance ($F_{resisting}$). In VisualFoundation, sliding forces can be applied to piers, walls, and grade beams. The sliding resistance consists of the friction resistance, the passive lateral resistance, the pile lateral resistance, and the specified external sliding resistance. The friction resistance is calculated as the net downward reaction force for the load combination multiplied by the specified coefficient of sliding friction. The total passive lateral resistance is calculated by the specified passive pressure multiplied by the total projected area of the footing in the horizontal direction under consideration. When piles are included in the model, an infinite pile resistance can be used (to disregard the sliding stability check) or the pile lateral resistance can be manually entered. The factor of safety (F.S.) for sliding stability is defined by the following equation:

F.S. sliding = $F_{resting}$ / $F_{sliding}$

2.9 Soil Support

Subgrade Modulus

One of the key material properties when designing a foundation is the coefficient of subgrade reaction (*k*), more commonly known as the soil subgrade modulus which in its simplest form is defined as follows:
$k = q / \Delta$

where:

q = load per unit area

Δ = settlement

The soil subgrade modulus is a measure of force per unit per length cubed which can be reduced to a spring constant given the area over which it acts. Therefore, spring supports can be added to a finite element mesh to simulate the reaction of the soil under a foundation. In VisualFoundation, the tributary area of the plates surrounding each node is multiplied by the subgrade modulus to determine the spring constant for each node in the finite element mesh. This is done on a per-node basis since the size of the plate elements surrounding each node can vary. Soil springs in VisualFoundation carry only compressive forces and have zero stiffness in tension. For a detailed discussion of subgrade modulus (specifically for foundation design), refer to the following:

- ACI Committee 336, "Suggested Analysis and Design Procedures for Combined Footings and Mats (ACI 336.2R-88)," American Concrete Institute, 1988
- Bowles, Joseph E., 1996. Foundation Analysis and Design, 5th Edition, McGraw-Hill Book Co. New York

The Portland Cement Association's "Concrete Floors on Ground" publication reports a subgrade modulus for different soil types. While these values are meant for slab-on-grade design and may not be appropriate for foundation design, they may provide a useful starting place. Site specific values, from a geotechnical engineer, will provide the most accurate solution. Due to the uncertain nature of soil parameters and since the analysis results might be sensitive to the input value of the subgrade modulus, it is wise to vary the subgrade modulus over a range of possible values during design.

Soil Bearing

Assuming a size (plan dimension and thickness) has been chosen to guarantee uplift and overturning stability, the next step is ensuring the size will not result in excessive bearing pressures between the footing and soil. Traditionally, this has been done assuming the footing is infinitely rigid relative to the soil. This results in a linearly varying bearing pressure when some overturning exists. For rectangular footings this bearing pressure calculation is not difficult. When the shape becomes more complex these calculations become much more tedious.

Many footings are of a thickness which causes them to have some flexibility relative to the soil and linearly varying bearing pressures do not exist. The interaction of soil stiffness and footing stiffness becomes an important analysis consideration. Use of the finite element method as employed by VisualAnalysis computational engine provides a fast and accurate way to deal with soil-structure interaction. VisualFoundation uses the finite element method creating a model of plate elements as well as beam or wall stiffening elements. Compression-only linear spring supports are used to model the soil. Accurate bearing pressures result from the analysis which can be compared to allowable or ultimate values specified by geotechnical engineers. In VisualFoundation, one bearing pressure value is manually specified in the **Project Manger | Modify** tab to use for unity checking. This value can be entered at either a service level or a strength level value.

3 Report

3.1 Reports

Reports in VisualFoundation are designed to present information in a clear, concise, and organized fashion. Reports can include both text-based and graphical information that can be printed to paper, to .pdf, or saved in a number of different file formats. Graphical information can be inserted into a report using the **Copy** and **Paste** commands or printed directly using the **File | Print** command.

Report Essentials

- Tables
- Saved Styles
- Grade Beam Graphs

Custom Report Logo

The report may be customized to include your own (company) logo in the header. All you need to do is create a logo image: ReportLogo.png or ReportLogo.jpg, and place it in the IES\Customer folder, which you can access via the **Tools | Custom Data** toolbar command. The image should be kept to less than 5 times wider than it is tall. It will be scaled to fit in the header area, but wide images may cause other text to start wrapping or get truncated. If the image works you'll see it in the report/preview immediately after restarting VisualFoundation. This feature is also available in other IES tools.

3.2 Tables

In VisualFoundation, tables are used to report information in a clear and concise manner. The tables available for the report are listed in the **Project Manager | Add Tables** tab when the Report View is active. Tables fall into one of six categories (Project, Structure, Load, Stability, Result, and Design) and will automatically appear or disappear depending on the items in the model (elements, loads, etc.) and the available analysis and design results. Hover the mouse over a table in the list to view its description.

Table Types

- **Project Tables** are used to document the project wide information for the model including the Project Settings, Service Load Cases, and Factored Load Combinations.
- **Structure Tables** are used to document the input data for various model objects including Slabs, Piers, Piles, Walls, Grade Beams, etc. Also, a Model Summary can be reported.
- **Load Tables** are used to document every load applied to the model in each service load cases including Area Loads, Pier Loads, Grade Beam Loads, and Wall Loads.
- **Stability Tables** are used to document the Overturning, Sliding, and Uplift Stability of the foundation system.
- **Result Tables** are used to document the analysis results for the elements in the model including Bearing Pressures and Displacements, Grade Beam Forces and Displacements, Pile Results, Slab Global Forces, etc.
- **Design Tables** are used to document the Design Settings, Design Summary, Shear Design, Punching Shear Design, Slab Flexure Design, Pier Design, Pile Design, Soil Bearing Capacity, etc. These tables are used to document the demand, capacity, unity checks, show intermediate calculation values, controlling code references, etc.

Adding & Removing Tables

To add a table to the report, simply ***drag*** the table from the **Project Manager | Add Tables** tab to the desired location in the report or ***double-click*** on the table to insert it at the end of the report. A list of the report's Included Tables is shown in the **Project Manager | Modify** tab which can be rearranged by ***dragging*** them with the mouse. To remove a table from the report, click the X next to the table in the Included Tables list or ***right-click*** on the table in the report and select Remove.

Modifying Tables

Tables can be modified using the Report Settings or Model Filters in the **Project Manager | Modify** tab. The Report Settings are used to specify which Service Cases and Result Cases to include in the report while the Model Filters are used to filter the items that are included in the report (such as Slabs, Piles, Piers, Grade Beams, etc.). Tables can also be modified by clicking on the tables in the report. ***Click*** the column header to sort the column, ***drag*** the column header to rearrange the columns in the tables, or ***drag*** the column boarders to adjust the column widths.

Selected Table

Click within a table to select the table and activate the **Project Manager | Selected Table** tab. In this tab, the Title can be modified, the columns can be sorted, and the page width can be defined. Choose which columns are included in the table under the Columns to Display section and drag the columns in this section to rearrange them in the report.

Selected Table Extremes

Certain tables have the Selected Table Extremes option available in the **Project Manager | Selected Table** tab. The following parameters are used to set how the information is filtered in the selected table.

- **Extreme Rows -** Set to show the Extreme Rows Only for the table or to Show All (which can lead to lengthy reports that may need to be filtered by result cases or reported items to be manageable).
- **Included Rows -** Specify how the extreme rows are considered.
    - **Max and Min -** Keep only the max and min values.
    - **Max -** Keep only the max value.
    - **Min -** Keep only the min value.
    - **Max/Min (when opposite sign)** - Keep the max and min values, if different signs, else keep the most extreme.
    - **Extreme -** Keep only the most extreme value, positive or negative.
- **Applies To -** Specify if the extreme rows be kept on a table wide basis or by each item in the table.
- **Consider Zero as Extreme -** Specify if zero should be considered an extreme value.
- **Show All Extreme Rows** - Choose to show all rows with the extreme value or only show the first occurrence of the extreme value.

Column Justification

Right click on a column in the report to set the Column Justification to Left, Center, or Right for an individual column in a table. In the Reports category of the Preferences, the default Justification for the Text data and Physical data can be specified.

3.3 Saved Reports

Both Project Reports and Report Styles can be created and saved in VisualFoundation. While Project Reports are saved in the .vfp project file, Report Styles are saved in the Customized Data Folder and can be used for multiple Projects. VisualFoundation also includes a few default IES Report Styles.

Project Reports

Save a Project Report

After creating a report, go to the **Project Manager | Reports** tab, name the report, and click the Save in Project button. This saves the Project Report in .vfp project file for easy access.

Delete a Project Report

To delete a Project Report, click the X next to the report in the **Project Manager | Reports** tab. The **Home | Undo** command can be used to restore a deleted report.

Report Styles

Save a Report Style

After creating a report, go to the **Project Manager | Reports** tab, name the report, and click the Save as Style button. This saves the Report Style in the Customized Data Folder in an XML file and makes the style available for use in other VisualFoundation projects.

Create a Report from a Style

To create a Report from Report Style, Simple ***double-click*** on the saved report in the **Project Manager | Reports** tab or ***drag*** the saved report onto the Report View.

Update a Report Style

To update a Report Style, create a report based on the style, modify the report as need, and save the style using the original name.

Delete a Report Style

To delete a style, click the X next to the style in the **Project Manager | Reports** tab or manually delete the style in the Customized Data Folder in an XML file. The only way to restore a style once it is deleted is to import a backup copy of the style file.

Share Report Styles

Report Styles can be shared by copying the XML style file (found in **Tools | Custom Data**) to the same location on another computer. The XML file can be manually edited to merge styles from other users. Always save a back up copy of the XML file before doing any manual customization so the file can be restored if it gets corrupted.

3.4 Grade Beam Graphs

Grade Beam Graphs display detailed diagrams (displacement, moment, shear, and torsion) for grade beams along their length. The results can be displayed for one single grade beam or for a chain of beams for a single Result Case.

Create a Single or Multi-Grade Beam Graph

To create a single grade beam graph, simply select a grade beam in the Model and Load, Analysis Results, or Foundation Design View and switch to the Grade Beam Graph view. A different grade beam can chosen from the dropdown in the **Project Manager | Graph Filter** tab. To create a multi-grade beam graph, simply select two or more connected grade

beams that form a line and switch to the Grade Beam Graph view. If the selected grade beams do not have local axes in the same direction, a note will appear on the graph indicating that the member chain has inconsistent local axes.

Customize a Grade Beam Graph

Grade beam graphs can be customized in the **Project Manager | Graph Filter** tab. Annotations, Data Points, Grid Lines, and Shadows can be adjusted for the plots. Use the Details Dialog to change specific graphic format information like colors and fonts as well as the basic type of plot used.

Print a Grade Beam Graph

Grade beam graphs can be printed directly using the **File | Print** command. The orientation (portrait or landscape) of the graphs can be set using the **File | Page Setup** dialog. Use **File | Print Preview** to view the page before printing.

Export a Grade Beam Graph

To export a Grade Beam Graph to another program, use the **Home | Copy** command to copy the Grade Beam Graph to the clipboard and then use **Paste** to insert the picture into the other application.

4 Integration

4.1 IES VisualAnalysis

There are two independent ways that VisualAnalysis and VisualFoundation can work together to help you solve problems. The tools are not completely integrated or interactive. Note: VisualFoundation and VisualAnalysis are sold separately by IES (Contact Sales for additional information). The Foundation Design Video in the VisualAnalysis help file shows how the programs work with each other.

1. Export from VisualAnalysis to VisualFoundation (using the **Export to VisualFoundation** feature).
2. Create a VisualAnalysis project file from VisualFoundation (using the **File | Export to VisualAnalysis** feature).

Foundation Design (Export from VisualAnalysis to VisualFoundation)

The **Export to VisualFoundation** feature exports the geometry and reactions from VisualAnalysis to set up a mat footing in VisualFoundation. This allows you to quickly generate a footing geometry for one or more columns from the VisualAnalysis project. The column locations are automatically imported and the reaction forces from VisualAnalysis service load cases are brought into VisualFoundation as loads. VisualFoundation provides foundation-specific design checks.

Foundation Design Procedure

1. Select one or more supported nodes in the VisualAnalysis project.
2. Choose **Structure | Create Foundation** to create a foundation for export.
3. Run an analysis, analyzing all the Service Load Case to obtain reaction forces at nodes.
4. With the foundation selected, use the **Tools | Export to VisualFoundation** feature to launch VisualFoundation and create a new footing project.
5. The VisualFoundation project can then be modified and design checks can then be performed. Save the project to create a .vfp file that can be opened for future use.
6. Upon exiting VisualFoundation, the foundation in VisualAnalysis is updated with a new boundary and thickness.
7. You can report a “Foundation Table" which lists all the footings by name, and provides thicknesses, areas, and which columns are supported. This table can be used to help coordinate with any saved VisualFoundation project files.

Foundation Design Limitations

- Support nodes must lie in a global plane (parallel to X, Y, or Z)
- Only member elements are exported as columns to VisualFoundation (walls or columns modeled out of plate elements do not export).
- The orientation of the model in VisualAnalysis is flexible (Y is the vertical axis by default). When the vertical axis is not Z, then model coordinates are mapped according to the following right hand rule transforms, and load cases are renamed accordingly.
    - Vertical Axis = X: Y-->X, Z-->Y, X-->Z
    - Vertical Axis = Y: X-->X, X-->Y, Y-->Z
- Multi-part footing or multi-thickness footings cannot be created directly from within VisualAnalysis
- The foundation cannot be edited or updated from VisualAnalysis (if you try to Design the foundation again it will simply create a new project in VisualFoundation).
- A VisualFoundation project cannot be opened and used to update a VisualAnalysis project (even if it was created from VisualAnalysis).

- If a multi-part footing is created in VisualFoundation, only the outside boundary is shown in VisualAnalysis.
- The details of the foundation are not shown or reported in VisualAnalysis (only the graphics and the thickness are shown and reported).
- VisualAnalysis is updated from VisualFoundation only when VisualFoundation is closed and only if it was launched from within VisualAnalysis.

VisualAnalysis Files (.vap) Created by VisualFoundation

VisualFoundation can be used to create a VisualAnalysis project file (.vap) using the **File | Export to VisualAnalysis** feature. The .vap file can be used to look at the finite element model that was created behind the scenes in VisualFoundation or to perform more advanced analyses (such as a dynamic analysis) which are not supported in VisualFoundation. VisualAnalysis also provides more advanced capabilities for designing grade beams and foundation elements.

4.2 IES VAConnect

Connection Design (Export from VisualFoundation to VAConnect)

VisualFoundation can export column-loads to VAConnect for a steel-column base plate design. After defining all the loads, select the pier(s) in the Model & Loads view, and use the **Foundation | Export to VAConnection** feature. Note: VAConnect is licensed separately. Contact Sales for additional information regarding this product.

5 Script

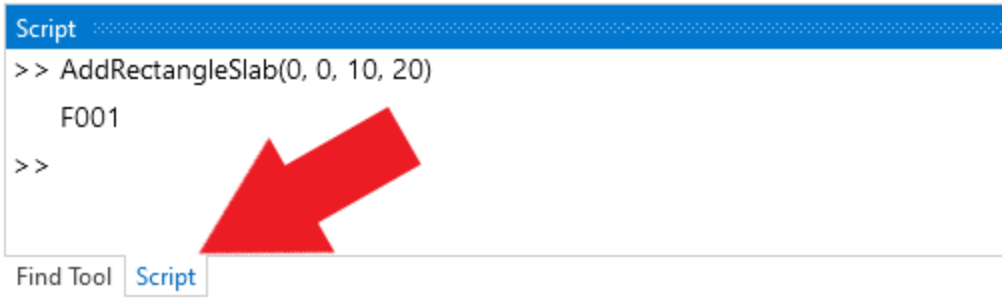
5.1 Script Overview

Introduction

The script feature in VisualFoundation is a powerful tool used to create models objects, generate service cases, apply loads, extract results, etc. using a command line interface. In addition to supporting the various Commands outlined in this Help File, the command line will accept any valid command in the C# Programming language (allowing the use of if statements, for loops, etc.). In addition to using the command line directly, more complex Scripts can be generated in any text file and read into the program. This allows models with complex geometry to be created, parametric studies to be performed, finite element meshes to be automatically refined until convergence, etc.

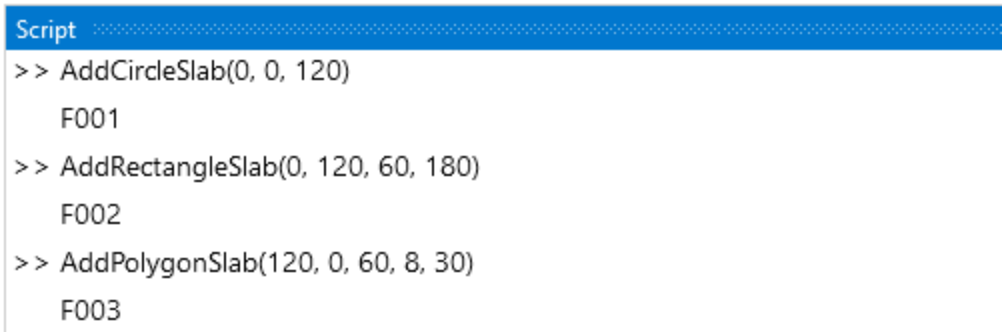
Command Line Basics

1. Location
    a. The Script tool is accessed by selecting the Script tab at the bottom of the Find Tool. By default, the Script will replace the Find Tool when selected, but the two can be floated and docked independently if needed.



2. Units
    a. Input and output are always in the current unit style. The style can be changed from the command line.

3. Add Slabs
    a. Add circular, rectangular, or polygon slabs of specified sizes and location using the appropriate commands.



    b. The return value can be suppressed by using a semi-colon on at the end of the command.

```
Script
>> AddCircleSlab(0, 0, 180);
>>
```

c. Information can be stored in variables and math can be performed in the command line. Variables circumvent the need to input the same number repeatedly. The example below uses variables to defined the center coordinates and the width of a rectangular slab (assuming the current unity style is Kips & Inches). Math is then used in the in the command line to define height of the slab as two times the width. Note: When storing a value in a variable, the line must end with a semicolon.

```
Script
>> var x = 72;
>> var y = 144;
>> var w = 24*12;
>> var h = 2*w;
>> AddRectangleSlab(x, y, w, h);
```

d. Slabs of non-standard shape can be defined based on lists of the X and Y coordinates or based on an array of Locations. Note: Locations can be stored as variables and used to define various items in the model, allowing the items to perfectly coincide.

```
Script
>> var xCoordinates = new List<double>(){0, 120, 0};
>> var yCoordinates = new List<double>(){0, 0, 120};
>> AddSlab(xCoordinates, yCoordinates);
>> var location1 = new Location(0, 0);
>> var location2 = new Location(-120, 0);
>> var location3 = new Location(0, 120);
>> AddSlab(location1, location2, location3);
```

e. The slab's reinforcement details (e.g. bar configuration, bar cover, design approach, etc.) can be defined using the various commands.

```
Script
>> SingleMat_Specify("F1", true, 2, "#6", 12, "#8", 16)
   Modified
>> DoubleMat_Optimize("F2", true, 2, true, 3, ("#4", 12), ("#5", 12))
   Modified
```

4. Add Piles

a. Piles can be added to the project at specific coordinates or Locations.

```
Script
>> AddPile(0, 0)
   P001
>> var location = new Location(12, 24);
>> AddPile(location)
   P002
```

b. Once the Pile is added, it will automatically be selected, and its properties can be modified in the Project

Manager as usual. Alternatively, most of the properties can be defined when adding Piles from the command line. Several options are available, see the Commands page for a complete list. Note: Item names must be enclosed in quotes.

```
Script
>> AddPile("pile", 12, 24, 48, "HP8X36", 15, "ASTM A992 Grade 50")
   pile
```

c. The PickMaterial(), PickDatabaseSection(), or PickParametricSection() calls can be used to set the material or cross-section when their exact name is unknown. These command calls bring up the material or shape dialog box and then return the name of the selected item.

```
Script
>> AddPile("pile", 12, 24, 48, PickDatabaseSection(), 15, PickMaterial())
   pile
```

d. The PickMaterial(), PickDatabaseSection(), or PickParametricSection() calls can also be used to set variables which can then be used when adding piles.

```
Script
>> var section = PickParametricSection();
>> var material = PickMaterial();
>> AddPile("pile", 12, 24, 48, section, 15, material)
   pile
```

5. Add Piers

a. Similar to Piles, Piers can be added to the project at specific coordinates or Locations.

```
Script
>> AddPier(0,0)
   C001
>> AddPier(new Location(12,24))
   C002
```

b. Piers can be renamed if the name that is automatically generated is not satisfactory.

```
Script
>> NamePier("C001", "C1")
   C1
```

c. There are various commands to modify the Pier parameters. See the Commands page for a complete list.

```
Script
>> RectanglePierSize("C1", 6, 12)
   Modified
>> SetPierHeight("C1", 16)
   Modified
>> SetPierSelfWeight("C1", true)
   Modified
>> RefinePier("C1", 6, 0.5, 4)
   Modified
```

6. Add Beams/Walls
    a. Beams and Walls can be added to the project based on start and end coordinates or locations.

```
Script
>> AddBeam(0, 0, 120, 240);
>> var startLocation = new Location(0, 0);
>> var endLocation = new Location(-120, -240);
>> AddWall(startLocation, endLocation);
```

    b. Various properties can be set when creating Beams or Walls.

```
Script
>> AddBeam("GB1", 0, 0, 120, 240, 24, 12, true);
>> AddWall("W1", 0, 0, -120, -240, 60, 12, true);
```

    c. A variety of parameters can be modified for existing beams or walls.

```
Script
>> SetBeamTopFlush("GB1");
>> WallSubgradeModulus("W1", 0.05);
```

    d. Beams and walls can be moved using the command line.

```
Script
>> MoveLineTo("GB1", 12, 24, 144, 288);
>> MoveLineBy("W1", 18, 36);
>> MoveStart("GB1", 0, 12);
>> MoveEnd("W1", new Location(-144, -360));
```

7. Apply Loads
    a. Rectangular, Circular, Tubular, and Rings loads can be added to the model with various input options (i.e. service case, location, pressure type, etc.). Note: Use a negative magnitude to apply loads in the negative global direction.

```
Script
>> RectangularLoad(0, 0, 12, 24, -50, 10, 20);
>> CircularLoad("D", 0, 0, 24, -50, 10, 20);
>> TubeLoad(0, 0, 24, 48, 6, -50, -100, 10, 20, true);
>> RingLoad("D", 0, 0, 36, 6, -50, -100, 10, 20, true);
```

b. Individual slabs or the full boundary can also be loaded using the command line.

```
Script
>> LoadSlab("F1", -50, 10, 20);
>> LoadFullBoundary("D", -50, -100, 10, 20, true);
```

c. A variety of options are available to apply loads to piers using the command line as outlined on the Commands page. In the example below, pier C1 is loaded in the D service case with a 5 kip sliding force in the X, a 10 kip sliding force in the Y, a 15 kip vertical force in the Z, a 100 kip-in moment about the X-axis, and a 200 kip-in moment about the Y-axis (assuming the current unity style is Kips & Inches).

```
Script
>> LoadPier("D", "C1", 5, 10, 15, 100, 200)
   Loaded C1
```

d. Both grade beams and walls are loaded using the LoadLine command which can take various inputs.

```
Script
>> LoadLine("GB1", 5, 10, 15, 100, 200, true, true);
>> LoadLine("D", "W1", 5, 10, 15, 100, 200, true, false);
```

8. Obtain Results

a. The maximum or minimum displacement, moment, shear, or bearing pressure across all result cases or for a specified result case can be obtained using command line. Also, the command line can output the displacement, moment, shear, or bearing pressure at a specific location for a defined result case.

```
Script
>> Bearing(true)
   330.924070043715
>> MomentX("1. D", false)
   -1.71614678087048
>> ShearY("5. D+0.6W »+X", 5.25, 5.0)
   -6.67649477643968
```

b. The maximum or minimum pile force across all result cases can be obtained using command line. Also, the command line can output the force in a specified pile for a defined result case.

```
Script
>> PileResult("P006", false)
   -22.0551072901692
>> PileResult("1. D", "P006")
   -14.7286060711977
```

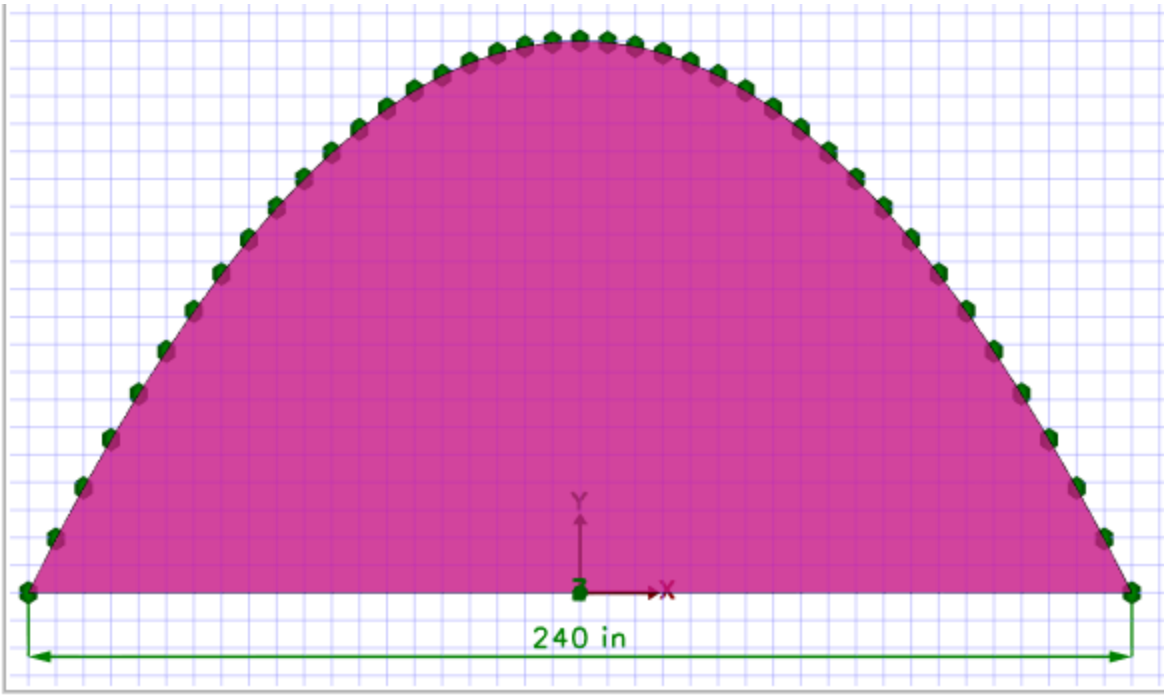
9. Miscellaneous

a. Integer Math: In the command line, the quotient of two integers is an integer which may not produce the intended result. See the C# Numeric Types Tutorial for more information.

```
Script
>> 1/2
   0
>> 1/2.0
   0.5
```

C# Programming

The command line accepts any valid command in the C# Programming Language, allowing the use of if statements, for loops, etc. In the example below, a command is defined to describe the y coordinate of a parabola terms of x. The Pow(Double, Double) method of the .Net System Math Class is used to perform squared operation. A four loop is used to add the x and y coordinates of the slab to lists which are then used to define the slab. In addition to using the command line to program directly, more complex Scripts can be generated in any text file and read into the program.




```
Script
>> double y(double x) => 120 - Math.Pow(x, 2)/120.0;
>> var xCoordinates = new List<double>();
>> var yCoordinates = new List<double>();
>> for(double x = -120; x <= 120; x = x + 6) {xCoordinates.Add(x); yCoordinates.Add(y(x));}
>> AddSlab(xCoordinates, yCoordinates)
   F001
```

5.2 Commands

Script Command Categories

- User Interface

	Command	Description	Example Input	Example Result
User Interface (BACK TO TOP)	Clear	Clears all text in the command line and clears any stored variables		
	Browse	Launches the Open dialog box to navigate to an external script to run		
	Up/Down Arrows	Use the "*Up Arrow*" and "*Down Arrow*" keys to navigate the command line history		
	Esc	Press the "*Esc*" key while a script is running to end the script run		
General (BACK TO TOP)	Help()	Launches the Help File and navigates to the command line overview		

	page		
Help(command)	Launches the Help File and navigates to the specified command	Help("AddSlab")	Launches the Help File and navigates to the AddSlab command
SetUnits(style)	Sets the unit style to a default or custom style in the program	SetUnits("Canadian")	Sets the programs unit style to Canadian
Select(names[])	Selects a specified item(s) (e.g. slab, vertex, etc.) in the model	Select("F1", "V1")	Selects slab F1 and vertex V1 in the model
Delete()	Deletes the selected model object(s) and/or load(s)	Select element Bm1, node N1, and vertex V1 then enter Delete()	Deletes element Bm1, node N1, and vertex V1 in the model
Delete(names[])	Deletes a specified item(s) (e.g. slabs, piers, piles, etc.) in the model	Delete("F1", "C1", "P1")	Deletes slab F1, pier C1, and pile P1 in the model
DeleteAll()	Deletes everything in the model		
Zoom(name)	Zooms to a specified item (e.g. slabs, piers, piles, etc.) in the model	Zoom("V1")	Zooms into vertex V1 in the model
Print(List<names>)	Prints the list of items in a comma-delimited format	Print(Slabs())	Prints the list of nodes in the project in a comma-delimited format
List(List<names>)	Lists the list of items with one item per line	List(Vertices())	Lists the list of vertices with one item per line
PickConcrete()	Opens the Concrete Material Database dialog box	SetPierMaterial("C1", PickConcrete())	Opens the Concrete Material Database dialog box to set the concrete for pier C1

	PickMaterial()	Opens the Material Database dialog box	AddPile("P1", 12, 24, 48, "HP8X36", 0, PickMaterial())	Opens the Material Database dialog box to define a material for pile P1 that is created at {12, 24}
	PickDatabaseSection()	Opens the Shape Database dialog box	AddPile("P1", 12, 24, 48, PickDatabaseSection(), 0, "ASTM A992 Grade 50")	Opens the Shape Database dialog box to define a database section for pile P1 that is created at {12, 24}
	PickParametricSection()	Opens the Parametric Shape Dimensions dialog box	AddPile("P1", 12, 24, 48, PickParametricSection(), 0, "Concrete (F'c = 4 ksi)")	Opens the Parametric Shape Dimensions dialog box to define a parametric section for pile P1 that is created at {12, 24}
Vertices (BACK TO TOP)	Vertices()	Returns a list of all the vertices in the project	Print(Vertices())	Prints the list of all the vertices in the project in a comma-delimited format
	MoveTo(name, X, Y)	Moves a specified vertex to a defined location	MoveTo("V1", 0, 0)	Moves vertex V1 to the origin
	MoveTo(name, location)	Moves a specified vertex to a predefined location	var location1 = new Location(10, 20); MoveTo("V1", location1)	Moves vertex V1 to location1
	MoveBy(name, distanceX, distanceY)	Moves a specified vertex by a specified distance in each global direction	MoveBy("V1", 5, 10)	Moves vertex V1 by 5 in the global X direction and 10 in the global Y direction
	X(name)	Returns the global X-coordinate of	X("V1")	Returns the global X-coordinate of

		the specified vertex		vertex V1
	Y(name)	Returns the global Y-coordinate of the specified vertex	Y("V1")	Returns the global Y-coordinate of vertex V1
Slabs (BACK TO TOP)	Slabs()	Returns a list of all the slabs in the project	Print(Slabs())	Prints the list of all the slabs in the project in a comma-delimited format
	AddSlab(List<X>, List<Y>)	Adds a slab defined by the a list of X coordinates and list of Y coordinates.	var X = new List<double>(){0, 1, 0}; var Y = new List<double>(){0, 0, 1}; AddSlab(X, Y)	Adds a slab defined by coordinate lists X and Y

	AddSlab(Location[])	Adds a slab defined by and array of locations	var location1 = new Location(0, 0);
			var location2 = new Location(1, 0);
			var location3 = new Location(0, 1);
			AddSlab(location1, location2, location3)
				Adds a slab defined by location1, location2, and location3

	AddRectangleSlab(centerX, centerY, width, height)	Adds a rectangular slab of specified width and height at a defined location	AddRectangleSlab(1, 2, 5, 15)	Adds a 5 wide by 10 tall rectangular slab centered at {1, 2}
	AddCircleSlab(centerX, centerY, radius)	Adds a circular slab of specified radius at a defined location	AddCircleSlab(1, 2, 5)	Adds a circular slab of radius 5 centered at {1, 2}
	AddPolygonSlab(centerX, centerX, radius, numberSides, angle)	Adds a polygon slab of specified radius at a defined location	AddPolygonSlab(1, 2, 5, 6, 45)	Adds a 6 sided polygon of radius 5 centered at {1, 2} with a rotation angle of 45
	ToggleHole(slab)	Toggles the Hole? parameter for the specified slab on or off	ToggleHole("F1")	Toggles the Hole? parameter on or off for slab F1
	SubgradeModulus(slab, modulus)	Sets the subgrade modulus to a	SubgradeModulus("F1", 0.25)	Sets the subgrade modulus for

		specified value for the defined slab		slab F1 to 0.25
	Thickness(slab, thickness)	Sets the thickness to a specified value for the defined slab	Thickness("F1", 12)	Sets the thickness for slab F1 to 12
	ToggleSelfWeight(slab)	Toggles the Add Self Weight parameter for the specified slab on or off	ToggleSelfWeight("F1")	Toggles the Add Self Weight parameter on or off for slab F1
	Height(slab, height)	Modifies the height of a rectangular or circular slab	Height("F1", 120)	Changes the height of slab F1 to 120
	Width(slab, width)	Modifies the width of a rectangular or circular slab	Width("F1", 120)	Changes the width of slab F1 to 120
	Theta(slab, theta)	Modifies the theta of a rectangular or polygon slab	Theta("F1", 45)	Changes theta of slab F1 to 45
	Radius(slab, radius)	Modifies the radius of a circular or polygon slab	Radius("F3", 120)	Changes the radius of slab F1 to 120
	SideCount(slab, sideCount)	Modifies the number of sides for a polygon slab	SideCount("F1", 6)	Changes the number of sides for slab F1 to 6
	SideLength(slab, length)	Modifies the side length for a polygon slab	SideLength("F1", 120)	Changes the side length for slab F1 to 120
	MoveCenterTo(slab, centerX, centerY)	Moves the center of the a circular, rectangular, or polygon slab to a defined location	MoveCenterTo("F1", 120, 240)	Moves the center of slab F1 to {120, 240}
	MoveCenterTo(slab, location)	Moves the center of the a circular, rectangular, or polygon slab to a defined location	var locaiton1 = new Location(120, 240); MoveCenterTo("F1", location1)	Moves the center of slab F1 to location1
	MoveCenterBy(slab, distanceX, distanceY)	Moves the center of the	MoveCenterBy("F1", 120, 240)	Moves the center of slab

	a circular, rectangular, or polygon slab by a specified amount		F1 by 120 in the X-direction and 240 in the Y-direction
SingleMat_Optimize(slab, xBarsTop, bottomCover, (string, double)[] barPatterns)	Sets the design approach for the slab to optimize, defines the reinforcement details, and sets the reinforcement search patterns for the single mat	SingleMat_Optimize("F1", true, 3, ("#4", 18), ("#4", 12), ("#5", 18), ("#5", 12))	Sets the design approach for slab F1 to optimize, defines the rebar directions and cover, and sets the search patterns as #4 @ 18, #4 @ 12, #5 @ 18, #4 @ 12, for the single mat
DoubleMat_Optimize(slab, topXTop, topCover, bottomXTop, bottomCover, (string, double)[] barPatterns)	Sets the design approach for the slab to optimize, defines the reinforcement details, and sets the reinforcement search patterns for both mats	DoubleMat_Optimize("F1", true, 2, true, 3, ("#4", 18), ("#4", 12), ("#5", 18), ("#5", 12))	Sets the design approach for slab F1 to optimize, defines the rebar directions and covers, and sets the search patterns as #4 @ 18, #4 @ 12, #5 @ 18, #4 @ 12, for both mats
SingleMat_Specify(slab, xBarsTop, bottomCover, xSize, xSpacing, ySize, ySpacing)	Sets the design approach for the slab to specify and defines the reinforcement mat and the reinforcement details	SingleMat_Specify("F1", true, 2, "#6", 12, "#8", 16)	Sets the design approach for slab F1 to specify and defines the single mat of reinforcement
DoubleMat_Specify(slab, topXTop, topCover, bottomXTop, bottomCover, topXSize, topXSpacing, topYSize, topYSpacing, bottomXSize, bottomXSpacing, bottomYSize, bottomYSpacing)	Sets the design approach for the slab to specify and defines the reinforcement mats (top and bottom) and	DoubleMat_Specify("F1", true, 2, true, 3, "#6", 12, "#6", 12, "#8", 16, "#8", 16)	Sets the design approach for slab F1 to specify and defines the two mats of reinforcement

		the reinforcement details		
Piles (BACK TO TOP)	Piles()	Returns a list of all the piles in the project	Print(Piles())	Prints the list of all the piles in the project in a comma-delimited format
	AddPile(X, Y)	Adds a default pile at a specified coordinate	AddPlie(0, 0)	Adds a default pile at the origin
	AddPile(location)	Adds a default pile at a specified location	var location1 = new Location(12, 24); AddPile(location1)	Adds a default pile at location1
	AddPile(name, X, Y, length, shape, theta, material)	Adds a pile with a defined name at a specified coordinates with the shape and material parameters defined	AddPile("P1", 12, 24, 48, "HP8X36", 15, "ASTM A992 Grade 50")	Adds pile P1 at {12, 24} that has a length of 48 with specified shape and material parameters
	AddPile(name, location, length, shape, theta, material)	Adds a pile with a defined name at a specified location with the shape and material parameters defined	var location1 = new Location(12, 24); AddPile("P1", location1, 48, "HP8X36", 15, "ASTM A992 Grade 50")	Adds pile P1 at location1 that has a length of 48 with specified shape and material parameters
	SetPileStiffness(pile, k)	Sets the stiffness for defined pile to a specific value	SetPileStiffness("P1", 1000)	Sets the stiffness of P1 to 1000
	SetPileRigid(pile)	Sets the specified pile to rigid	SetPileRigid("P1")	Sets pile P1 to rigid
	RefinePile(pile, offset, startLength, endLength)	Sets the Refine parameter for the specified pile to true and defines the mesh refinement parameters	RefinePile("P1", 6, 0.5, 4)	Sets the Refine parameter for pile P1 to true and set the offset to 6, the element length at start to 0.5, and the element length at end to 4

	UnRefinePile(pile)	Sets the Refine parameter for the specified pile to false	UnRefinePile("P1")	Sets the Refine parameter for pile P1 to false
	SetPileTensionOnly(pile)	Sets specified pile to Tension Only	SetPileTensionOnly("P1")	Sets pile P1 to Tension Only
	SetPileCompressionOnly(pile)	Sets specified pile to Compression Only	SetPileCompressionOnly("P1")	Sets pile P1 to Compression Only
	SetPileTwoWay(pile)	Sets specified pile to Tension and Compression	SetPileTwoWay("P1")	Sets pile P1 to Tension and Compression
	MovePileTo(pile, X, Y)	Moves specified pile to defined coordinates	MovePileTo("P1", 12, 24)	Moves pile P1 to {12, 24}
	MovePileTo(pile, location)	Moves specified pile to defined location	var location1 = new Location(12, 24); MovePileTo("P1", location1)	Moves pile P1 to location1
	MovePileBy(pile, distanceX, distanceY)	Moves specified pile by a specified distance in each global direction	MovePileBy("P1", 12, 24)	Moves pile P1 by 12 in the X-direction and 24 in the Y-direction
Piers (BACK TO TOP)	Piers()	Returns a list of all the piers in the project	Print(Piers())	Prints the list of all the piers in the project in a comma-delimited format
	AddPier(X, Y)	Adds a default pier at a specified coordinate	AddPier(0, 0)	Adds a default pier at the origin
	AddPier(location)	Adds a default pier at a specified location	var location1 = new Location(12, 24); AddPier(location1)	Adds a default pier at location1
	NamePier(currentName, newName)	Renames a pier	NamePier("C1", "C2")	Changes the name of pier C1 to C2
	SquarePierSize(pier, size)	Sets the specified pier shape to square and defines its size	SquarePierSize("C1", 8)	Sets pier C1 as a square and defines its size as 8
	RectanglePierSize(pier, sizeX, sizeY)	Sets the	RectanglePierSize("C1", 6, 12)	Sets pier C1

		specified pier shape to rectangular and defines its size		as a rectangle and defines its size as 6x12
	CirclePierSize(pier, radius)	Sets the specified pier shape to circle and defines its size	CirclePierSize("C1", 6)	Sets pier C1 as a circle and defines its radius as 6
	OctagonPierSize(pier, radius)	Sets the specified pier shape to octagon and defines its size	OctagonPierSize("C1", 9)	Sets pier C1 as an octagon and defines its radius as 9
	SetPierHeight(pier, height)	Sets the height for a specified pier	SetPierHeight("C1", 12)	Sets the height of pier C1 to 12
	SetPierMaterial(pier, material)	Sets the material for a specified pier	SetPierMaterial("C1", "Concrete (F'c = 4 ksi)")	Sets the material of pier C1 to Concrete (F'c = 4 ksi)
	SetPierTheta(pier, theta)	Sets the rotation for a specified pier	SetPierTheta("C1", 45)	Sets the rotation of pier C1 to 45
	SetPierSelfWeight(pier, includeWeight)	Includes or excludes the self weight for a specified pier	SetPierSelfWeight("C1", false)	Excludes the self weight for pier C1
	RefinePier(pier, offset, startLength, endLength)	Sets the Refine parameter for the specified pier to true and defines the mesh refinement parameters	RefinePier("C1", 6, 0.5, 4)	Sets the Refine parameter for pier C1 to true and set the offset to 6, the element length at start to 0.5, and the element length at end to 4
	UnRefinePier(pier)	Sets the Refine parameter for the specified pier to false	UnRefinePier("C1")	Sets the Refine parameter for pier P1 to false
	MovePierTo(pier, X, Y)	Moves specified pier to defined coordinates	MovePierTo("C1", 12, 24)	Moves pier C1 to {12, 24}
	MovePierTo(pier, location)	Moves	var location1 = new Location(12, 24);	Moves pier

		specified pier to defined location	MovePierTo("C1", location1)	C1 to location1
	MovePierBy(pier, distanceX, distanceY)	Moves specified pier by a specified distance in each global direction	MovePierBy("C1", 12, 24)	Moves pier C1 by 12 in the X-direction and 24 in the Y-direction
Grade Beams (BACK TO TOP)	Beams()	Returns a list of all the grade beams in the project	Print(Beams())	Prints the list of all the grade beams in the project in a comma-delimited format
	AddBeam(startX, startY, endX, endY)	Adds a grade beam between start coordinates and end coordinates	AddBeam(0, 0, 120, 240)	Adds a grade beam between {0, 0} and {120, 240}

	AddBeam(startLocation, endLocation)	Adds a grade beam between a start location and an end location	var location1 = new Location(0, 0);
			var location2 = new Location(120, 240);
			AddBeam(location1, location2)
				Adds a grade beam between location1 and location2

	AddBeam(name, startX, startY, endX, endY, depth, width, includeSelfWeight)	Adds a grade beam between start coordinates and end coordinates with a defined name, depth, , and width and with the self weight included or excluded	AddBeam("GB1", 0, 0, 120, 240, 24, 12, true)	Adds a grade beam named GB1 between {0, 0} and {120, 240} that is 24 deep, 12 wide, and includes its self weight

	AddBeam(name, startLocation, endLocation, depth, width, includeSelfWeight)	Adds a grade beam between a start location and an end location with a defined name, depth, and width and with the self weight included or excluded	var location1 = new Location(0, 0);
			var location2 = new Location(120, 240);
			AddBeam("GB1", location1, location2, 24, 12, true)
				Adds a grade beam named GB1 between location1 and location2 that is 24 deep, 12 wide, and includes its self weight

	BeamSubgradeModulus(beam, subgradeIncrement)	Sets the Subgrade	BeamSubgradeModulus("GB1", 0.1)	Sets the Subgrade

		Increment for the specified grade beam		Increment for grade beam GB1 to 0.1
	SetBeamTopFlush(beam)	Sets the vertical offset for the specified grade beam to Top Flush	SetBeamTopFlush("GB1")	Sets the vertical offset for grade beam GB1 to Top Flush
	SetBeamCentered(beam)	Sets the vertical offset for the specified grade beam to Centered	SetBeamCentered("GB1")	Sets the vertical offset for grade beam GB1 to Centered
	SetBeamBottomFlush(beam)	Sets the vertical offset for the specified grade beam to Bottom Flush	SetBeamBottomFlush("GB1")	Sets the vertical offset for grade beam GB1 to Bottom Flush
Walls (BACK TO TOP)	Walls()	Returns a list of all the walls in the project	Print(Walls())	Prints the list of all the walls in the project in a comma-delimited format
	AddWall(startX, startY, endX, endY)	Adds a wall between start coordinates and end coordinates	AddWall(0, 0, 120, 240)	Adds a wall between {0, 0} and {120, 240}

	AddWall(startLocation, endLocation)	Adds a wall between a start location and an end location	var location1 = new Location(0, 0);
			var location2 = new Location(120, 240);
			AddWall(location1, location2)
				Adds a wall between location1 and location2

	AddWall(name, startX, startY, endX, endY, height, thickness, includeSelfWeight)	Adds a wall between start coordinates and end coordinates with a defined name, height, and thickness and with the self weight included or excluded	AddWall("W1", 0, 0, 120, 240, 60, 12, true)	Adds a wall named W1 between {0, 0} and {120, 240} that is 60 tall, 12 thick, and includes its self weight

	AddWall(name, startLocation, endLocation, height, thickness, includeSelfWeight)	Adds a wall between a start location and an end location with	var location1 = new Location(0, 0);
			var location2 = new Location(120, 240);
			AddWall("W1", location1, location2, 60, 12, true)
				Adds a wall named W1 between location1 and location2 that

		a defined name, height, and thickness and with the self weight included or excluded		is 60 tall, 12 thick, and includes its self weight
	WallSubgradeModulus(wall, increment)	Sets the Subgrade Increment for the specified wall	WallSubgradeModulus("W1", 0.1)	Sets the Subgrade Increment for wall W1 to 0.1
Move Beam/Wall (BACK TO TOP)	MoveLineTo(beamOrWall, startX, startY, endX, endY)	Moves specified beam or wall to a defined start coordinate and end coordinate	MoveLineTo("L1", 0, 0, 120, 240)	Moves beam or wall L1 to span between coordinates {0, 0} and {120, 240}

	MoveLineTo(beamOrWall, startLocation, endLocation)	Moves specified beam or wall to a defined start location and end location	var location1 = new Location(0, 0);
			var location2 = new Location(120, 240);
			MoveLineTo("L1", location1, location2)
				Moves beam or wall L1 to span between location1 and locaiton2

	MoveLineBy(beamOrWallm, distanceX, distanceY)	Moves specified beam or wall by specified amounts in the X-direction and the Y-direction	MoveLineBy("L1", 12, 24)	Moves beam or wall L1 by 12 in the X-direction and 24 in the Y-direction
	MoveStart(beamOrWall, X, Y)	Moves the start point of a specified beam or wall to a defined coordinate	MoveStart("L1", 6, 12)	Moves the start coordinate of beam or wall L1 to {6, 12}
	MoveStart(beamOrWall, startLocation)	Moves the start point of a specified beam or wall to a defined location	var location1 = new Location(0, 0); MoveStart("L1", location1)	Moves the start coordinate of beam or wall L1 to locatoin1
	MoveEnd(beamOrWall, X, Y)	Moves the end point of a specified beam or wall to a defined coordinate	MoveEnd("L1", 6, 12)	Moves the end coordinate of beam or wall L1 to {6, 12}
	MoveEnd(beamOrWall, endLocation)	Moves the end point of a specified	var location1 = new Location(120, 120); MoveEnd("L1", location1)	Moves the end coordinate of

		beam or wall to a defined location		beam or wall L1 to locatoin1
Service Case (BACK TO TOP)	ServiceCases()	Returns a list of all the service cases in the project	Print(ServiceCases())	Prints the list of all the service cases in the project in a comma-delimited format
	AddServiceCase(name, source)	Adds a service case with a specified name and source	AddServiceCase("S-Unbalanced", "Snow")	Add service case S-Unbalanced with a Snow load source
	AddServiceCase(name, source, includeInCombos, pattern)	Adds a service case with a specified name and source and defines the pattern ID and if the case is included in the building code combinations	AddServiceCase("S-Unbalanced", "Snow", true, 1)	Add service case S-Unbalanced with a Snow load source and includes the service case in the building code combination and sets the pattern ID to 1
	VerticalSelfWeight(name)	Turns on the self weight for a specified service case	VerticalSelfWeight("D")	Turns on the self weight for service case D
	NoSelfWeight(name)	Turns off the self weight for a specified service case	NoSelfWeight("D")	Turns off the self weight for service case D
Region Loads (BACK TO TOP)	RectangularLoad(X, Y, widthX, widthY, UniformPressure, MX, MY)	Applies a widthX by widthY rectangular load centered at {X, Y} with a specified uniform pressure with MX and MY overturning loads in the current service case	RectangularLoad(0, 0, 2, 4, -50, 10, 20)	Applies a 2 by 4 rectangular load centered at {0, 0} with a specified uniform pressure of -50 with 10 and 20 overturning loads in the current service case
	RectangularLoad(ServiceCase, X, Y, widthX, widthY, UniformPressure, MX, MY)	Applies a widthX by widthY rectangular load centered	RectangularLoad("D", 0, 0, 2, 4, -50, 10, 20)	Applies a 2 by 4 rectangular load centered at {0, 0} with a specified

		at {X, Y} with a specified uniform pressure with MX and MY overturning loads in the defined service case		uniform pressure of -50 with 10 and 20 overturning loads in the "D" service case
	RectangularLoad(X, Y, widthX, widthY, startPressure, endPressure, MX, MY, linearX)	Applies a widthX by widthY rectangular load centered at {X, Y} with a specified linear pressure with MX and MY overturning loads in the current service case	RectangularLoad(0, 0, 2, 4, -50, -100, 10, 20, true)	Applies a 2 by 4 rectangular load centered at {0, 0} with a linear pressure varying from -50 to -100 in the X-direction with 10 and 20 overturning loads in the current service case
	RectangularLoad(serviceCase, X, Y, widthX, widthY, startPressure, endPressure, MX, MY, linearX)	Applies a widthX by widthY rectangular load centered at {X, Y} with a specified linear pressure with MX and MY overturning loads in the defined service case	RectangularLoad("D", 0, 0, 2, 4, -50, -100, 10, 20, true)	Applies a 2 by 4 rectangular load centered at {0, 0} with a linear pressure varying from -50 to -100 in the X-direction with 10 and 20 overturning loads in the "D" service case
	CircularLoad(X, Y, radius, uniformPressure, MX, MY)	Applies a circular load with a specified radius centered at {X, Y} with a specified uniform pressure with MX and MY overturning loads in the current service case	CircularLoad(0, 0, 4, -50, 10, 20)	Applies a circular load with a 4 radius centered at {0, 0} with a -50 uniform pressure with 10 and 20 overturning loads in the current service case
	CircularLoad(serviceCase, X, Y, radius, uniformPressure, MX, MY)	Applies a circular load with a specified	CircularLoad("D", 0, 0, 4, -50, 10, 20)	Applies a circular load with a 4 radius

		radius centered at {X, Y} with a specified uniform pressure with MX and MY overturning loads in the defined service case		centered at {0, 0} with a -50 uniform pressure with 10 and 20 overturning loads in the "D" service case
	CircularLoad(X, Y, radius, startPressure, endPressure, MX, MY, linearX)	Applies a circular load with a specified radius centered at {X, Y} with a specified linear pressure with MX and MY overturning loads in the current service case	CircularLoad(0, 0, 4, -50, -100, 10, 20, true)	Applies a circular load with a 4 radius centered at {0, 0} with a -50 to -100 linear pressure in the X-direction with 10 and 20 overturning loads in the current service case
	CircularLoad(serviceCase, X, Y, radius, startPressure, endPressure, MX, MY, linearX)	Applies a circular load with a specified radius centered at {X, Y} with a specified linear pressure with MX and MY overturning loads in the defined service case	CircularLoad("D", 0, 0, 4, -50, -100, 10, 20, true)	Applies a circular load with a 4 radius centered at {0, 0} with a -50 to -100 linear pressure in the X-direction with 10 and 20 overturning loads in the "D" service case
	TubeLoad(X, Y, widthX, widthY, thickness, uniformPressure, MX, MY)	Applies a widthX by widthY tube load centered at {X, Y} with a specified uniform pressure with MX and MY overturning loads in the current service case	TubeLoad(0, 0, 2, 4, 0.5, -50, 10, 20)	Applies a 2 by 4 tube load centered at {0, 0} with a -50 uniform pressure with 10 and 20 overturning loads in the current service case
	TubeLoad(serviceCase, X, Y, widthX, widthY, thickness, uniformPressure,	Applies a widthX by	TubeLoad("D", 0, 0, 2, 4, 0.5, -50, 10, 20)	Applies a 2 by 4 tube load

	MX, MY)	widthY tube load centered at {X, Y} with a specified uniform pressure with MX and MY overturning loads in the defined service case		centered at {0, 0} with a -50 uniform pressure in the X-direction with 10 and 20 overturning loads in the "D" service case
	TubeLoad(X, Y, widthX, widthY, thickness, startPressure, endPressure, MX, MY, linearX)	Applies a widthX by widthY tube load centered at {X, Y} with a specified linear pressure with MX and MY overturning loads in the current service case	TubeLoad(0, 0, 2, 4, 0.5, -50, -100, 10, 20, true)	Applies a 2 by 4 tube load centered at {0, 0} with a -50 to -100 linear pressure in the X-direction with 10 and 20 overturning loads in the current service case
	TubeLoad(serviceCase, X, Y, distanceX, distanceY, thickness, startPressure, endPressure, MX, MY, linearX)	Applies a widthX by widthY tube load centered at {X, Y} with a specified linear pressure with MX and MY overturning loads in the defined service case	TubeLoad("D", 0, 0, 2, 4, 0.5, -50, -100, 10, 20, true)	Applies a 2 by 4 tube load centered at {0, 0} with a -50 to -100 linear pressure in the X-direction with 10 and 20 overturning loads in the "D" service case
	RingLoad(X, Y, radius, thickness, uniformPressure, MX, MY)	Applies a ring load with a specified radius and thickness centered at {X, Y} with a specified uniform pressure with MX and MY overturning loads in the current service case	RingLoad(0, 0, 4, 1, -50, 10, 20)	Applies a ring load with a 4 radius and 1 thickness centered at {0, 0} with a -50 uniform pressure with 10 and 20 overturning loads in the current service case
	RingLoad(serviceCase, X, Y, radius, thickness, uniformPressure, MX, MY)	Applies a ring load with a specified radius and	RingLoad("D", 0, 0, 4, 1, -50, 10, 20)	Applies a ring load with a 4 radius and 1 thickness

		thickness centered at {X, Y} with a specified uniform pressure with MX and MY overturning loads in the defined service case		centered at {0, 0} with a -50 uniform pressure with 10 and 20 overturning loads in the "D" service case
	RingLoad(X, Y, radius, thickness, startPressure, endPressure, MX, MY, linearX)	Applies a ring load with a specified radius and thickness centered at {X, Y} with a specified linear pressure with MX and MY overturning loads in the current service case	RingLoad(0, 0, 4, 1, -50, -100, 10, 20, true)	Applies a ring load with a 4 radius and 1 thickness centered at {0, 0} with a -50 to -100 linear pressure in the X-direction with 10 and 20 overturning loads in the current service case
	RingLoad(serviceCase, X, Y, radius, thickness, startPressure, endPressure, MX, MY, linearX)	Applies a ring load with a specified radius and thickness centered at {X, Y} with a specified linear pressure with MX and MY overturning loads in the defined service case	RingLoad("D", 0, 0, 4, 1, -50, -100, 10, 20, true)	Applies a ring load with a 4 radius and 1 thickness centered at {0, 0} with a -50 to -100 linear pressure in the X-direction with 10 and 20 overturning loads in the "D" service case
	LoadSlab(uniformPressure, MX, MY)	Applies a uniform pressure with MX and MY overturning loads to the selected slab in the current service case	LoadSlab(-50, 10, 20)	Applies a -50 uniform pressure with 10 and 20 overturning loads to the selected slab in the current service case
	LoadSlab(slab, uniformPressure, MX, MY)	Applies a uniform pressure with MX and MY overturning loads to the	LoadSlab("F1", -50, 10, 20)	Applies a -50 uniform pressure with 10 and 20 overturning loads to slab

		specified slab in the current service case		F1 in the current service case
	LoadSlab(serviceCase, slab, uniformPressure, MX, MY)	Applies a uniform pressure with MX and MY overturning loads to the specified slab in the defined service case	LoadSlab("D", "F1", -50, 10, 20)	Applies a -50 uniform pressure with 10 and 20 overturning loads to slab F1 in the "D" service case
	LoadFullBoundary(uniformPressure, MX, MY)	Applies a uniform pressure with MX and MY overturning loads to the full boundary in the current service case	LoadFullBoundary(-50, 10, 20)	Applies a -50 uniform pressure with 10 and 20 overturning loads to the full boundary in the current service case
	LoadFullBoundary(serviceCase, uniformPressure, MX, MY)	Applies a uniform pressure with MX and MY overturning loads to the full boundary in the defined service case	LoadFullBoundary("D", -50, 10, 20)	Applies a -50 uniform pressure with 10 and 20 overturning loads to the full boundary in the "D" service case
	LoadFullBoundary(startPressure, endPressure, MX, MY, linearX)	Applies a linear pressure with MX and MY overturning loads to the full boundary in the current service case	LoadFullBoundary(-50, -100, 10, 20, true)	Applies a -50 to -100 linear pressure in the X-direction with 10 and 20 overturning loads to the full boundary in the current service case
	LoadFullBoundary(serviceCase, startPressure, endPressure, MX, MY, linearX)	Applies a linear pressure with MX and MY overturning loads to the full boundary in the defined service case	LoadFullBoundary("D", -50, -100, 10, 20, true)	Applies a -50 to -100 linear pressure in the X-direction with 10 and 20 overturning loads to the full boundary in the "D" service case
Pier Loads [(BACK TO TOP)](#)	LoadPier(FX, FY, FZ, MX, MY)	Applies specified forces and moments to the selected	LoadPier(5, 10, 15, 100, 200)	Applies forces and moments to the selected pier in the current

		pier in the current service case		service case
	LoadPier(pier, FX, FY, FZ, MX, MY)	Applies specified forces and moments to the defined pier in the current service case	LoadPier("C1", 5, 10, 15, 100, 200)	Applies forces and moments to pier "C1" in the current service case
	LoadPier(serviceCase, pier, FX, FY, FZ, MX, MY)	Applies specified forces and moments to the defined pier in the specified service case	LoadPier("D", "C1", 5, 10, 15, 100, 200)	Applies forces and moments to pier "C1" in the D service case
Beam/Wall Loads (BACK TO TOP)	LoadLine(FX, FY, FZ, MX, MY, distributed = false, local = false)	Applies specified forces and moments to the selected grade beam or wall in the current service case as a resultant or distributed load in the global or local direction	LoadLine(5, 10, 15, 100, 200)	Applies forces and moments to the selected grade beam or wall in the current service case. The loads are applied as resultants in the global direction since "distributed" and "local" are false by default.
	LoadLine(item, FX, FY, FZ, MX, MY, distributed = false, local = false)	Applies specified forces and moments to the defined grade beam or wall in the current service case as a resultant or distributed load in the global or local direction	LoadLine("GB1", 5, 10, 15, 100, 200, true, true)	Applies forces and moments to grade beam GB1 in the current service case. The loads are applied as distributed loads in the local direction.
	LoadLine(serviceCase, item, FX, FY, FZ, MX, MY, distributed = false, local = false)	Applies specified forces and moments to the defined grade beam or wall in the	LoadLine("D", "W1", 5, 10, 15, 100, 200, true, false)	Applies forces and moments in the D service case to the grade beam GB1 in the current

		defined service case as a resultant or distributed load in the global or local direction		service case. The loads are applied as distributed loads in the global direction.
Analysis Settings (BACK TO TOP)	SetMeshCoarse()	Sets the Mesh Refinement to Coarse		
	SetMeshMedium()	Sets the Mesh Refinement to Medium		
	SetMeshFine()	Sets the Mesh Refinement to Fine		
	SetMeshCustom(elementCount)	Sets the Mesh Refinement to User Defined and sets the Element Count to a specified value	SetMeshCustom(6000)	Sets the Mesh Refinement to User Defined and sets the Element Count to 6000
	SetEFactor(factor)	Sets the Concrete Elastic Modulus Factor in the project	SetEFactor(2)	Sets the Concrete Elastic Modulus Factor to 2 in the project
	ThicknessOverlap(setting)	Changes the Thickness Overlap setting to the specified value	ThicknessOverlap("Smallest")	Changes the Thickness Overlap setting to Smallest Thickness
	SoilOverlap(setting)	Changes the Soil Modulus Overlap setting to the specified value	SoilOverlap("Largest")	Changes the Soil Modulus Overlap setting to Largest Thickness
	SetMaterial()	Opens the Material Database dialog box to select the concrete material that is used for the slabs and beams in the project		
	SetMaterial(concrete)	Sets the concrete	SetMaterial("Concrete (F'c = 3.5 ksi)")	Sets the concrete

		material that is used for the slabs and beams in the project		material that is used for the slabs and beams in the project to Concrete (F'c = 3.5 ksi)
Stability Settings (BACK TO TOP)	CheckStabilityService()	Performs the stability checks at the service level		
	CheckStabilityStrength()	Performs the stability checks at the strength level		
	OverturningFS(FS)	Sets the overturning stability factor of safety to the specified value	OverturningFS(2.5)	Sets the overturning stability factor of safety to 2.5
	UpliftFS(FS)	Sets the uplift stability factor of safety to the specified value	UpliftFS(2.0)	Sets the uplift stability factor of safety to 2.0
	SlidingFS(FS)	Sets the sliding stability factor of safety to the specified value	SlidingFS(2.0)	Sets the sliding stability factor of safety to 2.0
	ReduceSeismicOVT(reduce)	Sets the Reduce Seismic Overturning parameter to yes or no which reduces the seismic overturning by 25% per ASCE 7	ReduceSeismicOVT(true)	Sets the Reduce Seismic Overturning parameter to yes
	SlidingResistanceX(resistance)	Sets the additional sliding resistance force for sliding stability checks in the X-direction to a specified	SlidingResistanceX(10)	Sets the additional sliding resistance force for sliding stability checks in the X-direction to 10

		value		
	SlidingResistanceY(resistance)	Sets the additional sliding resistance force for sliding stability checks in the Y-direction to a specified value	SlidingResistanceY(20)	Sets the additional sliding resistance force for sliding stability checks in the Y-direction to 20
Soil Settings [(BACK TO TOP)](#)	CheckBearingService()	Performs the bearing checks at the service level		
	CheckBearingStrength()	Performs the bearing checks at the strength level		
	SetBearingPressure(pressure)	Sets the bearing pressure to a specified value	SetBearingPressure(2000)	Sets the bearing pressure to 2000
	SetFriction(coefficient)	Sets the coefficient of sliding friction between the bottom of the footing and the soil to a specified value	SetFriction(0.45)	Sets the coefficient of sliding friction between the bottom of the footing and the soil to 0.45
	SetPassivePressure(pressure)	Sets the average passive pressure across the thickness of the footing to a specified value	SetPassivePressure(50)	Sets the average passive pressure across the thickness of the footing to 50
Status [(BACK TO TOP)](#)	Status(itemTitle)	Returns the Project Status of the specified item. -1 = Failing 0 = Not found or not available 1 = Has a warning	Status("Pier Capacity")	Returns the Project Status of the Pier Capacity

		2 = Ok		
Pipeline (BACK TO TOP)	Meshing()	Pauses the script until the meshing completes. Note: Only used for external scripts and must be preceded by "await"	await Meshing();	Pauses the external script until the meshing completes
	Stability()	Pauses the script until the stability checks complete. Note: Only used for external scripts and must be preceded by "await"	await Stability();	Pauses the external script until the stability checks complete
	Analysis()	Pauses the script until the analysis completes. Note: Only used for external scripts and must be preceded by "await"	await Analysis();	Pauses the external script until the analysis completes
	Design()	Pauses the script until the design completes. Note: Only used for external scripts and must be preceded by "await"	await Design();	Pauses the external script until the design completes
Result Cases (BACK TO TOP)	Results()	Returns a list of all the result cases in the project	Print(Results())	Prints the list of all the result cases in the project in a comma-delimited format
Plate Results (BACK TO TOP)	Displacement(max)	Returns the maximum or minimum displacement	Displacement(false)	Returns the minimum displacement in the global

		in the global Z-direction across all result cases		Z-direction across all result cases
	Displacement(result, max)	Returns the maximum or minimum displacement in the global Z-direction for the defined result case	Displacement("1. D", true)	Returns the maximum displacement in the in the global Z-direction for the "1. D" result case
	Displacement(result, X, Y)	Returns the displacement in the global Z-direction at a specified location for the defined result case	Displacement("2. D+L", 6, 12)	Returns the displacement in the global Z-direction at {6, 12} for the "2. D+L" result case
	ShearX(max)	Returns the maximum or minimum shear force that acts on the global X-face of the plate elements (VX) across all result cases	ShearX(false)	Returns the minimum shear force that acts on the global X-face of the plate elements (VX) across all result cases
	ShearX(result, max)	Returns the maximum or minimum shear force that acts on the global X-face of the plate elements (VX) for the defined result case	ShearX("1. D", true)	Returns the maximum shear force that acts on the global X-face of the plate elements (VX) for the defined result case
	ShearX(result, X, Y)	Returns the shear force that acts on the global X-face of the plate elements (VX) at a specified location for the defined result case	ShearX("2. D+L", 6, 12)	Returns the shear force that acts on the global X-face of the plate elements (VX) at {6, 12} for the "2. D+L" result case
	ShearY(max)	Returns the maximum or	ShearY(true)	Returns the maximum

		minimum shear force that acts on the global Y-face of the plate elements (VY) across all result cases		shear force that acts on the global Y-face of the plate elements (VY) across all result cases
	ShearY(result, max)	Returns the maximum or minimum shear force that acts on the global Y-face of the plate elements (VY) for the defined result case	ShearY("1. D", false)	Returns the minimum shear force that acts on the global Y-face of the plate elements (VY) for the "1. D" result case
	ShearY(result, X, Y)	Returns the shear force that acts on the global Y-face of the plate elements (VY) at a specified location for the defined result case	ShearY("2. D+L", 6, 12)	Returns the shear force that acts on the global Y-face of the plate elements (VY) at {6, 12} for the "2. D+L" result case
	MomentX(max)	Returns the maximum or minimum bending moment that acts on the global X-face of the plate elements (MX) across all result cases	MomentX(true)	Returns the maximum bending moment that acts on the global X-face of the plate elements (MX) across all result cases
	MomentX(result, max)	Returns the maximum or minimum bending moment that acts on the global X-face of the plate elements (MX) for the defined result case	MomentX("1. D", false)	Returns the minimum bending moment that acts on the global X-face of the plate elements (MX) for the "1. D" result case
	MomentX(result, X, Y)	Returns the bending	MomentX("2. D+L", 6, 12)	Returns the bending

		moment that acts on the global X-face of the plate elements (MX) at a specified location for the defined result case		moment that acts on the global X-face of the plate elements (MX) at {6, 12} for the "2. D+L" result case
	MomentY(max)	Returns the maximum or minimum bending moment that acts on the global Y-face of the plate elements (MY) across all result cases	MomentY(false)	Returns the minimum bending moment that acts on the global Y-face of the plate elements (MY) across all result cases
	MomentY(result, max)	Returns the maximum or minimum bending moment that acts on the global Y-face of the plate elements (MY) for the defined result case	MomentY("1. D", true)	Returns the maximum bending moment that acts on the global Y-face of the plate elements (MY) for the "1. D" result case
	MomentY(result, X, Y)	Returns the bending moment that acts on the global Y-face of the plate elements (MY) at a specified location for the defined result case	MomentY(result, X, Y)	Returns the bending moment that acts on the global Y-face of the plate elements (MY) at a {6, 12} for the "2. D+L" result case
	MomentXY(max)	Returns the maximum or minimum twisting moment (MXY) across all result cases	MomentXY(true)	Returns the maximum twisting moment (MXY) across all result cases
	MomentXY(result, max)	Returns the maximum or minimum twisting	MomentXY("1. D", false)	Returns the minimum twisting moment

		moment (MXY) for the defined result case		(MXY) for the ""1. D"" result case
	MomentXY(result, X, Y)	Returns the twisting moment (MXY) at a specified location for the defined result case	MomentXY("2. D+L", 6, 12)	Returns the twisting moment (MXY) at a {6, 12} for the "2. D+L" result case
	Bearing(max)	Returns the maximum or minimum bearing pressure across all result cases	Bearing(false)	Returns the minimum bearing pressure across all result cases
	Bearing(result, max)	Returns the maximum or minimum bearing pressure for the defined result case	Bearing("1. D", true)	Returns the maximum bearing pressure for the "1. D" result case
	Bearing(result, X, Y)	Returns the bearing pressure at a specified location for the defined result case	Bearing("2. D+L", 6, 12)	Returns the bearing pressure at a {6, 12} for the "2. D+L" result case
Pile Results (BACK TO TOP)	PileResult(pile, max)	Returns the maximum or minimum force in the specified pile across all result cases	PileResult("P1", false)	Returns the minimum force in pile P1 across all result cases
	PileResult(result, pile)	Returns the force in the specified pile for the defined result case	PileResult("1. D", "P1")	Returns the force in pile P1 for the "1. D" result case
Report (BACK TO TOP)	AddTable(title)	Adds a specified table to the report	AddTable("Sliding Stability")	Adds the Sliding Stability table to the report
	ExportReport(path)	Exports the report to a specified path	ExportReport("C:/Users/your.login/Desktop/Report.pdf")	Saves the report as a .pdf on the desktop for the specified user

5.3 External Scripts

Running External Scripts

1. Opening External Scripts
    a. Type Browse into the command line to launch the Open dial box and select the script text file to use.
    b. Directly type the path of the script text file into to the command line and press Enter.
2. Example Scripts
    a. Combined Footing
    b. Model & Load
    c. Refine Mesh
    d. Optimize Bearing

5.4 Example: Combined Footing

Combined Footing

```
//Builds a Combined Footing
SetUnits("Kips & Feet");
DeleteAll();

//Add Slabs
AddRectangleSlab(0, 0, 4, 12);
var w = 12;
var h = 8;
var l1 = new Location(0, -0.5*h);
var l2 = new Location(0, 0.5*h);
var l3 = new Location(w, 0.25*h);
var l4 = new Location(w, -0.25*h);
AddSlab(l1, l2, l3, l4);
MoveCenterTo("F001", -2, 0);

//Add Piers
AddPier(2, 0);
AddPier(l4);
MovePierBy("C002", -2, 2);
CirclePierSize("C001", 0.5);
CirclePierSize("C002", 0.5);
SetPierMaterial("C001", "Concrete (F'c = 4 ksi)");
SetPierMaterial("C002", PickConcrete());

//Add Wall
AddWall("W1", -2, -6, -2, 6, 4, 8/12.0, true);

//Add Semicircular Slab
double x(double y) => -Math.Sqrt(Math.Pow(6, 2) - Math.Pow(y, 2)) - 4;
var xCoordinates = new List<double>();
var yCoordinates = new List<double>();
for(double y = -6; y <= 6; y = y + 0.25) {xCoordinates.Add(x(y)); yCoordinates.Add(y);}
AddSlab(xCoordinates, yCoordinates);

//Add Loads
LoadLine("D", "W1", 0, 0, -50, 0, 0);
```

```
LoadPier("D", "C001", 0, 0, -20, 0, 0);
LoadPier("D", "C002", 0, 0, -20, 0, 0);
LoadPier("W+X", "C001", 0, 0, 25, 0, 0);
LoadPier("W+X", "C002", 0, 0, -25, 0, 0);

//Obtain Results
await Analysis();
$"Max Bearing = {Bearing(true)}; Max Bearing Dead Load = {Bearing("1. D", true)};
Bearing Dead Load at (0, -6) = {Bearing("1. D", 0, -6)}"
```

5.5 Example: Model & Load

Model & Load

```
//Builds a slab and grade beam system supported by piles that is
//loaded by four column piers, by a uniform slab load, and
//by line loads on the perimeter grade beams.

//Clear project & set units
DeleteAll();
SetUnits("USA (Mixed)");

//Project settings
SetMeshMedium();
SetMaterial("Concrete (F'c = 3.5 ksi)");
SetBearingPressure(2000);

//Project parameters to define the size of the square slab (a)
//and the extent of the grade beams (b)
var a = 25;
var b = 0.5*a-2;

//Add slabs
var s1 = AddRectangleSlab(0, 0, a, a);
Thickness(s1, 12);
SingleMat_Specify(s1, true, 5, "#6", 12, "#6", 12);

//Define locations and store as variables which are used to define
//piers and grade beams so that they perfectly align
var lP1 = new Location(-b, b);
var lP2 = new Location(0, b);
var lP3 = new Location(b, b);
var lP4 = new Location(-b, 0);
var lP5 = new Location(0, 0);
var lP6 = new Location(b, 0);
var lP7 = new Location(-b, -b);
var lP8 = new Location(0, -b);
var lP9 = new Location(b, -b);

//Add piles
AddPile(null, lP1, 8, "HP8X36", 0, "ASTM A992 Grade 50");
AddPile(null, lP2, 8, "HP8X36", 0, "ASTM A992 Grade 50");
AddPile(null, lP3, 8, "HP8X36", 0, "ASTM A992 Grade 50");
```

```
AddPile(null, lP4, 8, "HP8X36", 0, "ASTM A992 Grade 50");
AddPile(null, lP5, 8, "HP8X36", 0, "ASTM A992 Grade 50");
AddPile(null, lP6, 8, "HP8X36", 0, "ASTM A992 Grade 50");
AddPile(null, lP7, 8, "HP8X36", 0, "ASTM A992 Grade 50");
AddPile(null, lP8, 8, "HP8X36", 0, "ASTM A992 Grade 50");
AddPile(null, lP9, 8, "HP8X36", 0, "ASTM A992 Grade 50");

//Add grade beams
AddBeam("GBX1", lP1, lP3, 12.0, 10.0, true);
AddBeam("GBX2", lP4, lP6, 16.0, 10.0, true);
AddBeam("GBX3", lP7, lP9, 12.0, 10.0, true);
AddBeam("GBY1", lP1, lP7, 12.0, 10.0, true);
AddBeam("GBY2", lP2, lP8, 16.0, 10.0, true);
AddBeam("GBY3", lP3, lP9, 12.0, 10.0, true);

//Add piers
var c1 = AddPier(-0.5*b, 0);
var c2 = AddPier(0.5*b, 0);
var c3 = AddPier(0, 0.5*b);
var c4 = AddPier(0, -0.5*b);
SquarePierSize(c1, 9);
SquarePierSize(c2, 9);
SquarePierSize(c3, 9);
SquarePierSize(c4, 9);

//Add slab loads
LoadSlab("D", s1, -25, 0, 0);

//Add perimeter grade beam loads
LoadLine("D", "GBX1", 0, 0, -10, 0, 0);
LoadLine("D", "GBX3", 0, 0, -10, 0, 0);
LoadLine("D", "GBY1", 0, 0, -10, 0, 0);
LoadLine("D", "GBY3", 0, 0, -10, 0, 0);
LoadLine("W+X", "GBY1", 0, 0, 5, 0, 0);
LoadLine("W+X", "GBY3", 0, 0, -5, 0, 0);
LoadLine("W+X", "GBX1", 0, 0, 0, 0, 50);
LoadLine("W+X", "GBX3", 0, 0, 0, 0, 50);

//Add pier loads
LoadPier("D", c1, 0, 0, -5, 0, 0);
LoadPier("D", c2, 0, 0, -5, 0, 0);
LoadPier("D", c3, 0, 0, -5, 0, 0);
LoadPier("D", c4, 0, 0, -5, 0, 0);
LoadPier("W+X", c1, 0, 0, 15, 0, 0);
LoadPier("W+X", c2, 0, 0, -15, 0, 0);

//Set view
Zoom(s1)
```

5.6 Example: Refine Mesh

Refine Mesh

```
//Use a loop to refine the mesh of a spread footing
```

```
SetUnits("Kips & Inches");
DeleteAll();

//Build the model
AddRectangleSlab(0, 0, 48, 48);
var pier = AddPier(0, 0);
var columnSize = 12.0;
SquarePierSize(pier, columnSize);

//Load the pier
LoadPier("D", pier, 0.0, 0.0, -20.0, 0.0, 0.0);

//set a coarse mesh for most of the footing
SetMeshCustom(100);

//Try finer and finer around the pier until
//the moment at the face of the pier stops changing

var elementSize = 5.0; //start with a 5 inch element around the pier
var delta = 100.0;  //initialize to a big value
var moment = 100.0; //initialize to a big value
int i = 0; //keep track of the number of iterations needed to find the converged value
while(delta > 0.05)
{
    i++;
    elementSize *= 0.7;
    RefinePier(pier, columnSize + 3.0, elementSize, elementSize * 5.0);
    await Analysis();
    var newMoment = MomentX("1. 1.4D", columnSize / 2.0, 0.0);
    delta = Math.Abs(moment - newMoment);
    moment = newMoment;
}

//Print out the final results
$"Mux = {moment}; Final Delta = {delta}; Trials = {i}"
```

5.7 Example: Optimize Bearing

Optimize Bearing

```
//Optimize the footing size for bearing

SetUnits("Kips & Inches");
DeleteAll();

//Build the model
var radius = 12.0;
var slab = AddCircleSlab(0, 0, radius);
var pier = AddPier(0, 0);
SetMeshCoarse();

//Load the pier
LoadPier("D", pier, 0.0, 0.0, -3.0, 0.0, 0.0);
LoadPier("L", pier, 0.0, 0.0, -6.0, 0.0, 2.0);

//Add one inch to the footing radius until the soil bearing checks pass
```

```
await Design();
while(Status("Soil Bearing") < 2)
{
    radius += 1.0;
    Radius(slab, radius);
    await Design();
}

//Print out the final results
$"Final Radius = {radius}"
```