VisualPlate 7.0

User's Guide

Updated: 12/27/2024

VisualPlate 7.0 User's Guide

Table of Contents

1 Introduction

1.1 Welcome to VisualPlate 7.0

VisualPlate will help you solve a wide variety plate bending problems using any material. This powerful analysis program is designed to be flexible so that you can efficiently solve your unique problem. It automates much of the work involved in finite element modeling that can be tedious in a more general tool. The software will analyze your model to determine stresses, moments, shears, and displacements using finite element analysis.

Getting Started

- Use **File | Open Example Project** 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

VisualPlate 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 ensure 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 VisualPlate on your computer, you become a registered user of the software. The VisualPlate 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 it 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 VisualPlate may be purchased directly from IES, Inc.

IES, Inc.

Integrated Engineering Software, Inc.
3740 Equestrian Ln Ste 1
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 geometries
- Create any plate geometry by sketching boundaries
- Import from CAD for complex and detailed boundaries
- Work with column-lines to organize and modify structures
- Create plates with various thicknesses
- Define line stiffeners (e.g. beams) with various shapes and material properties
- Define point supports (e.g. columns)
- Define line supports (e.g. walls)
- Easily refine finite element mesh
- Create new model objects by copying existing items

Loading

- Model objects can be loaded in multiple service load cases (e.g. Dead, Live, etc.)
- Automatic building code load combinations are available
- Includes IBC, ASCE, and NBC Load Combinations
- Customizable building code combinations (see Load Case Manager)
- Create custom load combinations in any project
- Apply point loads, line/distributed loads, and area load pressures to the structure
- 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
- Advanced error-checking and reporting
- Export FEA model to VisualAnalysis to better understand model details or to perform a more sophisticated analysis

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 perform member design or check design specifications
- Does not produce structural drawings
- Cannot model a system of disconnected plates.

Be a Squeaky Wheel

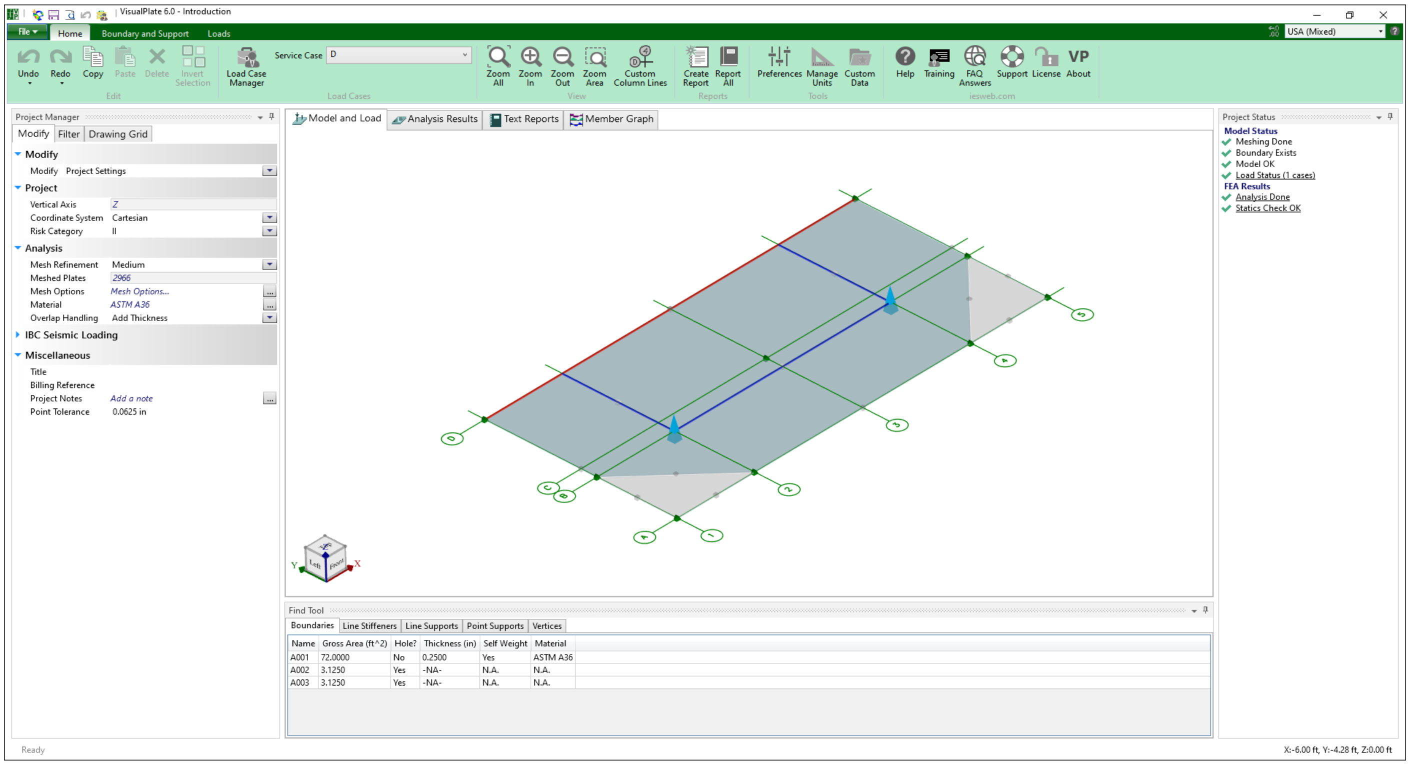
If you need a new feature, please let us know. We are always looking for ways to improve products in ways that you desire. See Technical Support.

1.3 Program Layout

The best way to learn VisualPlate 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.



Project Manager

Modify | Filter | Drawing Grid

Modify

Modify	Project Settings

Project

Vertical Axis	*Z*
Coordinate System	Cartesian
Risk Category	II

Analysis

Mesh Refinement	Medium
Meshed Plates	*2966*
Mesh Options	*Mesh Options...*
Material	*ASTM A36*
Overlap Handling	Add Thickness

IBC Seismic Loading

Miscellaneous

Title	
Billing Reference	
Project Notes	*Add a note*
Point Tolerance	0.0625 in

Project Status

Model Status
- ✓ Meshing Done
- ✓ Boundary Exists
- ✓ Model OK
- ✓ Load Status (1 cases)

FEA Results
- ✓ Analysis Done
- ✓ Statics Check OK

Find Tool

Boundaries | Line Stiffeners | Line Supports | Point Supports | Vertices

Name	Gross Area (ft^2)	Hole?	Thickness (in)	Self Weight	Material
A001	72.0000	No	0.2500	Yes	ASTM A36
A002	3.1250	Yes	-NA-	N.A.	N.A.
A003	3.1250	Yes	-NA-	N.A.	N.A.

Main Menu / Toolbar

The main menu, Toolbar, or Ribbon, contains various commands to direct VisualPlate. 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 VisualPlate. 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.
- 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 Analysis 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, 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 Member Graphs.

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 and analysis progress. Background processing is done on a separate thread of your processor so you may continue working while the program runs. 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 results 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 boundaries, 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 (boundary, load point, point support, 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.

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
- ***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
- **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 boundaries, line supports, line stiffeners, 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.

Context Menu:

Right-Click the mouse for a short menu of relevant commands based on the view and what is selected.

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 7.0 (January 2025)

Scripts & Command Line

- Command Line is a powerful new way to drive the program:
    - Definite slabs, line stiffeners, point supports, line supports etc.
    - Apply loads to regions, load points, and line stiffeners
    - Adjust the analysis settings
    - Extract plate results (displacements, shears, moments, etc.)
    - Add tables to report and export reports
- External Script files can automate common tasks and much more:
    - Generate boundaries, apply loads, extract results
    - Refine the mesh until model converges
    - Perform model optimization

New Report Features

- Added option to save reports in the project
- Added option to save reports as a style
- Reorganized report project manager
- Table of Contents report table
- Added Clear Tables button

Other Features

- Query plate results at a specified location
- Improved behavior of support springs
- Updated Microsoft .NET framework for features and performance

1.5 Release History

Overview

Versions

- Version 7.0 released January 2025
- Version 6.0 released April 2022
- Version 5.0 released November 2018
- Version 4.0 released December 2017
- Version 3.0 released May 2015
- Version 2.0 released January 2014
- Version 1.0 released January 2011

Version 7

Scripts & Command Line

- Command Line is a powerful new way to drive the program:
    - Definite slabs, line stiffeners, point supports, line supports etc.
    - Apply loads to regions, load points, and line stiffeners
    - Adjust the analysis settings
    - Extract plate results (displacements, shears, moments, etc.)
    - Add tables to report and export reports
- External Script files can automate common tasks and much more:
    - Generate boundaries, apply loads, extract results
    - Refine the mesh until model converges
    - Perform model optimization

New Report Features

- Added option to save reports in the project
- Added option to save reports as a style
- Reorganized report project manager
- Table of Contents report table
- Added Clear Tables button

Other Features

- Query plate results at a specified location
- Improved behavior of support springs
- Updated Microsoft .NET framework for features and performance

Version 6

New Features

- Boundary's mesh can be refined at load point and point support locations
- Improved handling of models with misaligned geometry
- Improved drawing grids
- Improved the performance of generated loads
- Program ensures area loads are properly modeled before starting the analysis
- Line stiffener loads remain constant when switching between resultant and distributed
- Load Case Manager columns now retain their size and order
- Plate result diagram overhaul
- Improved filtering of table extremes
- Preferences created for justification of text and data in reports
- Individual column justification added in reports (right click on column)

Fixes & Minor Changes

- Improved Help File documentation
- Load Area information added to Area Uniform Loads tab in the Find Tool
- Added Sigma Top columns to the Plate Stresses table

- Crash recovery file improvements
- Updated the c++ runtimes to the current standards

Version 5

General

- Graphic wire-frame in picture view mode
- Project Manager: Categories remember last open/collapsed state
- Project Manager: drop-lists are activated by clicking anywhere, not just on arrow
- Improved warning message formatting
- Improved Print Preview display for graphics
- Column-line graphics are "filtered" to avoid clutter
- Improved area load graphics with shaded tops
- Graphics performance is **up to 20x faster**
- Removed 'memory leaks', which slowed program over time

Modeling

- Full area support (rigid and flexible) can now be applied to an area
- Ability to insert a vertex along an area side
- DXF import shows bounds and an option for centering at the origin
- Polygon boundary defined by side length in addition to radius

Loading

- Ability to move area loads by side coordinates
- Line stiffener loads can now be applied relative to wall orientation (parallel and perpendicular)
- Multiple selection in Load Case Manager

Analysis

- Line supports now insert a "drilling" spring support
- Individual service cases can be analyzed
- Analysis is now **12% faster**

Reporting

- **Name filters** are now enabled for both graphic and report filters
- Unavailable tables shown disabled with reason for not being available
- Table drop position used to locate when adding new tables in Text Reports
- Full report can now have categories filtered (model, loads, results, graphics)

Version 4

The concrete design module of VisualPlate 3.0 was removed and ConcreteBending 4.0 was created

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 boundaries
- Easier to edit
- Automatic split/connect meshing for crossing or overlapping line supports, beams
- Control over overlapping for plate thicknesses
- DXF import has insertion point so huge coordinate files can be moved.

Loading

- Implements ASCE 7 load combinations
- More Direct full-boundary loading

Analysis

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

Reporting

- Powerful Report Viewer
- Many tables and options
- Saved reports in project files
- Paste graphics into reports
- Complete project report
- Predefined, customizable reports

1.6 Preferences

VisualPlate 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 VisualPlate 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

The project preferences affect new projects, and do not affect the current project. For current projects, use the settings in **Project Manager**.

- **History Files** - Set how many once-per-day backup files VisualPlate should keep. Files are located in **Home | Custom Data** in the History Projects folder.
- **Next Inspector Field On Enter** - In the Modify tab, 'Enter' can simply accept changes or also advance like a 'Tab' to the next row.

Data

Set the default name prefixes for line stiffeners, line supports, and areas. These settings apply to the first objects created in a new project or immediately after restarting VisualPlate.

Fonts

Change the character size and styles used to display text in graphic views and reports.

Reports

- **Member Results Offsets** - Input the number of interpolated result locations along the member in some results tables (2-201).
- **Customer Logo** - Specify the location of a logo to use for the reports. If left blank, AppData\Local\IES\Customer\ReportLogo.jpg will be tried.
- **Logo Alignment** - Select the alignment of the logo in the header.
- **Maximum Page Count** - Input the number of pages allowed in a report before a warning and truncation occurs.
- **Header Height** - Set the height of the header in the report.
- **Justification, Text Data** - Specify the justification of the text data in the report tables.
- **Justification, Physical Data** - Specify the justification of the physical data in the report tables.

Graphics

- **Graphic Sizes** - Change how large objects are drawn in graphic views.
- **Rotate, Pan, & Zoom -** Control how much or how fast the view changes with mouse-wheel or arrow keys.
- **Default Snap Points** - Set the default number of evenly spaced internal points that can be snapped to.
- **Print Resolution** - Set the DPI (dots per inch) precision to be used when displaying graphics views on the printer or when placing graphics information on the clipboard.

Colors

Change the colors of objects in Graphic Views and Reports. Every visible object type shown in graphic views have a default color (e.g. line stiffeners are blue by default).

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.

2 Modeling

2.1 Boundaries

Plate boundaries are the primary elements modeled and analyzed in VisualPlate. Boundaries with a wide variety of shapes and sizes can be modeled in the program. The boundary can be supported by Point Supports and Line Support and stiffness can be added by modeling Line Stiffeners into the boundary. A single material is used for all boundaries in the project and is defined in the **Project Settings**.

Modeling

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

Holes

A selected boundary 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 boundaries are considered (i.e. the loading that occurs over holes is not distributed to adjacent plate elements)

Loading

Boundaries may be loaded with Full Area, Circular, Rectangular, Tubular, and Ring area loads or by concentrated Point loads. Loads can also be transferred to the slab by Line Stiffeners.

Analysis

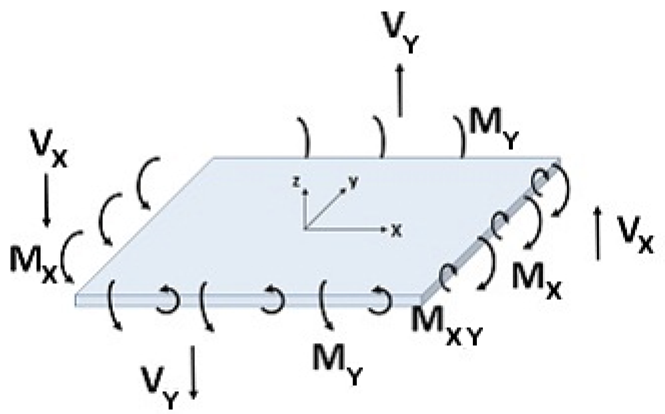
Finite element analysis is approximate and numerical. Use mesh refinement and examine results carefully, before trusting analysis results.

FEA Model

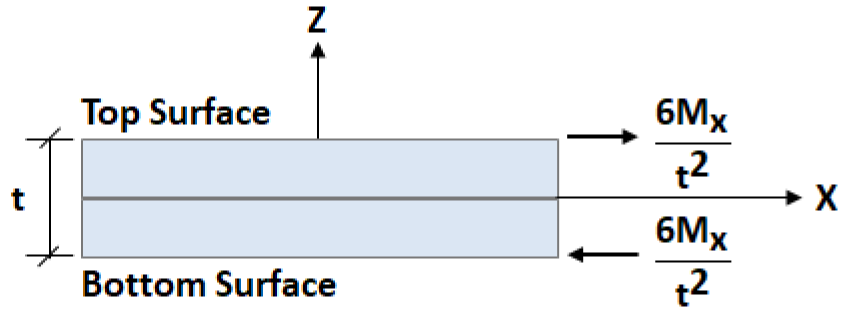
Boundaries in VisualPlate are meshed into triangular FEA plate elements. See the Analysis page for more information on the FEA mesh density, including the available settings and options. For a better understanding of the FEA model that is built by VisualPlate, use the **File | Export a VA Project** feature to create a VisualAnalysis project and examine the FEA model in more detail. More information on integration between VisualPlate and other IES tools can be found on the Integration page 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. Tensile stresses are positive and compressive stresses are negative.



Plates are only subject to bending stresses in VisualPlate. Therefore, the mid-plane is the neutral axis and the normal stresses at this plane are zero. The stresses at the top and bottom surfaces are equal in magnitude and opposite in sign.



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**.

Plate Diagrams

Plate result moment, shear, and displacement diagrams are available in the Analysis Results view. Left-click and drag a line across the slab to view the result at the line's location or right-click and select Show Plate Result Diagram from the context menu to view the diagrams for the selected result case. The slice direction and location can be adjusted within the Plate Result Diagrams dialog box along with the number of plots displayed and the plot type. Note: The result location and plot settings will persist within one session of VisualPlate. This allows the Plate Result Diagrams dialog box to be closed to select a different result case or to modify the model without losing the adjustments that were made in the dialog.

Reporting Analysis Results

Analysis results can also be viewed in text form using the Text Reports tab. Once on the Text Report tab, a list of available 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 view, and using the right-click context menu to *Report*

Selected.

2.2 Point Supports

Point Supports in VisualPlate are used to support the plate at a specified location in the model.

Modeling

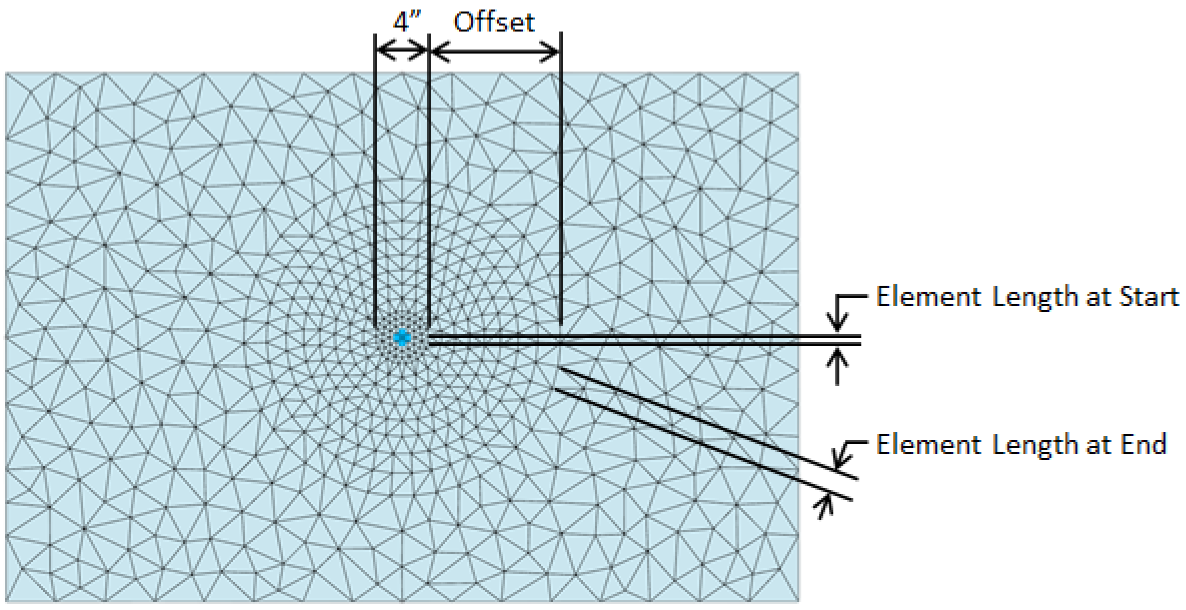
Point Supports are defined by their location in the XY-plane in the model. Note: In the Finite Element Analysis (FEA) model, only a single node in the slab's mesh is restrained at a point support; therefore, care should be taken when using a point supports to model a physical support that has a large cross-sections.

Support Type

VisualPlate only allows deflection in the Z-direction and rotation about the X-axis and Y-axis. Therefore, Point Supports have the option of being set to Fixed, Free, or Specified-K (stiffness) for Force-Z, Moment-X, and Moment-Y.

Mesh Refinement

The FEA method is approximate and the accuracy of the 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 point supports where the support can create stress concentrations). Select a point support(s) and set Refine? = True in the **Project Manager | Modify** tab to refine the mesh in the regions of the chosen point support(s). The mesh refinement parameters for point supports are defined in the image below. Note: Since point supports are a single point in the model and do not have physical sized, the offset and element length at start begin at a 4 inch diameter circle that encloses the point support.



Point Support Mesh Refinement

Loading

Point Supports cannot be directly loaded in VisualPlate. A Load Point can be defined at the same location as the point support (i.e. connected to the same node in the finite element model) and load can be applied to the Load Point.

Analysis

The force in the Z-direction and the moments in the X-direction and the Y-direction are calculated for each load case and reported for the Point Supports in VisualPlate. The displacement in the Z-direction and the rotations about the X-axis and the Y-axis are also reported.

Reports

Report tables, which include modeling and result information for the Point Supports, are available in the Text Report view.

2.3 Line Supports

Line Supports are used in VisualPlate to restrain the model along a series of nodes that fall on a straight line which is defined by a start coordinate and an end coordinate in the XY-plane.

Support Properties

Line Supports only provide restraint along a single line of nodes in the FEA model. VisualPlate only allows deflection in the Z-direction and rotation about the X-axis and Y-axis. Line Supports fix the deflection of the nodes on the line in the Z-direction and fix the rotation of the nodes about a line in the XY-plane perpendicular to the Line Support. The three support options (Pinned, Fixed, and Spring) allow the rotational restraint about the line defined by the Line Support in the XY-plane to be specified.

Analysis Results

The total force in the Z-direction and the total moment about the Line Support is calculated and reported for each load case.

2.4 Line Stiffeners

Modeling

Line Stiffeners are created using the **Boundary and Support | Draw Line Stiffeners** command to draw line stiffeners on top of an existing plate boundary in the model. Line stiffeners distribute applied loads to the plate boundary and add stiffness to the model. Material Properties are defined for the selected line stiffener(s) in the **Project Manager | Modify** tab. Line stiffeners 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 line stiffeners).

Shape Properties

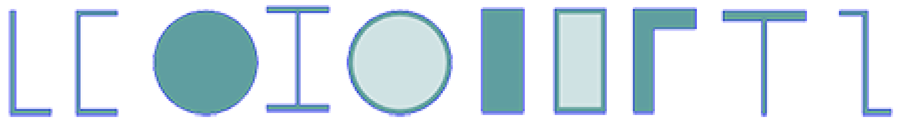
The shape for a line stiffener member is defined on the **Project Manager | Modify** tab when one or more stiffener is selected. There are four types of shapes available for use in VisualPlate: Database Shapes, Standard Parametric Shapes, Custom Shapes, and Analysis Blobs. Custom Shapes are created in IES ShapeBuilder and added them to the Custom Shape Database.

Database Shapes

IES includes a large shape database of steel, wood, aluminum, cold-formed, and other shapes common in the USA and some other countries. The database is customizable using IES ShapeBuilder, but cannot be modified directly using VisualPlate. Selecting a database shape will typically also define material and therefore it is best to define the shape prior to defining the material. The shape database contains Virtual Joists and Virtual Joist Girders which are developed by the Steel Joist Institute. Their website has information on the basic concept and purpose. You may create models with these shapes as if they were steel beams (please understand their purpose and limitations before using them).

Standard Parametric Shapes

Parametric shapes are defined by dimensions such as width, depth, and thickness, which are input by the user. Parametric shapes are commonly used for defining concrete members (e.g. square, rectangle, round, etc.). VisualPlate offers the following types of parametric shapes: Angle, Channel, Circle, I-Shape, Pipe, Rectangle, Rectangular Tube, Spandrel, Tee, and Zee.



Custom "Blobs"

Custom Blobs are a quick way to defined the numerical properties of a shape that are needed to perform the analysis (the actual dimensions of the shape are not defined). Custom Blobs are only available in the project in which they were created (i.e. they are not added to the shape database). Use Custom Blobs with care as the shape properties are not checked other than to ensure they are positive.

Quick Shape Pick List

The **Project Manager | Modify** tab shows the types of shapes available in the Source drop-down list: Standard Parametric, Database Shape, <Add Custom 'Blob'...>, <ShapeBuilder>, followed by a list of shapes that are already used or have been recently used in the project. Use this list to quickly find a shape for a new line stiffeners.

Loading

Vertical forces and moments in two directions can be applied to line stiffeners. The self weight of the line stiffener can be included by checking the option for the selected stiffener in the **Project Manager | Modify** tab. Line stiffener loads can be entered as distributed evenly along the member length or as the resultant total. Loads can be entered in the global coordinate system or in the line stiffener's local coordinate system (defined as parallel or perpendicular to the member).

Analysis

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

Displaying Results Graphically

The line stiffener results from the finite element analysis are displayed graphically in the Analysis Results view. The **Project Manager | Results** tab displays the numerical results that correspond to the colored graphics. The line stiffener's displacement, shear force, moment, and stresses can be viewed by changing the Result Type drop-down menu under the Line Stiffener Results category of the **Project Manager | Result Filter** tab. With nothing selected, the results displayed in the **Project Manager | Results** tab are a summary for the selected result case, whereas if a single line stiffener is selected, the Project Manager shows the result range for the selected line stiffener. 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 line stiffener(s) for each result case can be viewed using the Member Graph tab.

Reporting Line Stiffener Results

Analysis Results can be viewed in text form using the Text Reports tab. Once on the Text Report tab, a list of available 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 Load Combination Criteria

Load Combination Sets

Several sets of building code load combinations are built into VisualPlate. Custom load combinations may also be created in the Load Combinations tab of the Load Case Manager. Service level load combinations may be added if you want to look at results.

Importing Load Combinations

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

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. It is important to note that these parameters (such as SDS and SD1) 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. VisualPlate does not automatically generate any loads, just the combination cases.

2.6 Loads

In VisualPlate, loads are applied to the model in a service load case. The appropriate service case can be selected in the Service Case drop-down list in the Home or Loads ribbon. Loads may be easier to see and select by rotating the view to get a 3D perspective of the model. Note: VisualPlate only allows deflection in the Z-direction and rotation about the X-axis and Y-axis.

Concentrated Loads

Concentrated loads are applied to Load Points by selecting the Load Point(s) and clicking the Apply Load button in the Loads ribbon. The concentrated load consists of a force in the global Z-direction and moments in the global X-direction and Y-direction.

Line Loads

Line loads are applied to Line Stiffeners by selecting the Line Stiffener(s) and clicking the Apply Load button in the Loads ribbon. The line load consists of a force in the global Z-direction and/or moments which can be defined in the global X-direction and Y-direction or defined parallel and perpendicular to the line stiffener (i.e. in the line stiffener's local direction). Line loads can be entered as resultants or as distributed loads in the model.

Area Loads

Partial Boundary Loads

Rectangular, circular, tubular, and ring loads are used to apply pressure loads and/or overturning moments to some portion of the Boundary in the model. Pressures may be uniform or linearly varying in the global X-direction or global Y-direction. First set the Area Load type in the Loads ribbon and then sketch the load on the drawing grid in the model. The size and location of the load can be modified if needed after the load has been generated. Note: Loads will not be applied to holes in the plate's boundary.

Complete Boundary Loads

Select one or more Boundary and use the Apply Load command in the Loads ribbon to load the selected boundary. Alternatively, the Apply Multiple Boundary Load command in the Loads ribbon can be use to create a load that is applied to all boundaries in the model. Note: Boundary loads will automatically adjust to load the entire boundary if the boundaries's size or location is modified.

2.7 Self Weight

In VisualPlate, the Z-axis is defined as the vertical axis. Therefore, gravity can only act perpendicular to the plate boundaries defined in the model. The self-weight of the elements can be set on an individual basis.

Plates and Line Stiffeners

To include a plate or line stiffener's weight in the Dead-Load service case, switch to the Model and Load view, select the object, and check the Add Self Weight box in the Project Manager | Modify tab. The item's density is determined by the selected material.

Point Supports, Line Supports, and Load Points

These items do not have self-weight.

2.8 Load Points

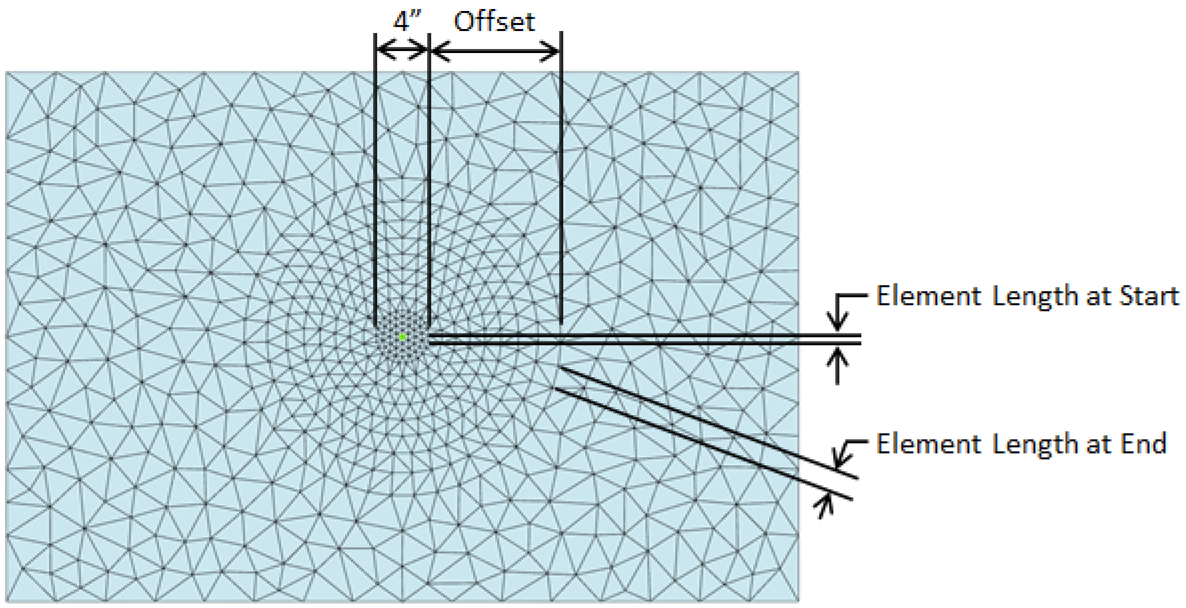
Load Points in VisualPlate are used to apply concentrated loads and moments to plates at a specified location.

Modeling

Load points are defined by their location in the model. Note: In the Finite Element Analysis (FEA) model, a single node in the slab's mesh is loaded; therefore, care should be taken when using a load point to model a physical load that has a large cross-sections.

Mesh Refinement

The FEA method is approximate and the accuracy of the 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 load points where concentrated loads can create stress concentrations). Select a load point(s) and set Refine? = True in the **Project Manager | Modify** tab to refine the mesh in the regions of the chosen load point(s). The mesh refinement parameters for load points are defined in the image below. Note: Since load points are a single point in the model and do not have physical sized, the offset and element length at start begin at a 4 inch diameter circle that encloses the load point.



Load Point Mesh Refinement

Loading

Concentrated forces (compression and tension), in addition to moments in two directions can be applied to load points.

Analysis

During the finite element analysis each load point is modeled as a single node connected to the plate's finite element mesh. Therefore, care should be taken when using a load point to model a physical load that has a large cross-sections.

Report

Multiple report tables, which include modeling, loading, and result information for the Load Points, are available in the Text Report view.

2.9 Material Properties

The material properties for plates and line stiffeners are defined using materials from either the IES Material Databases or the Custom Material Database as discussed below.

IES Material Databases

Several common material databases are included with VisualPlate listed under IES in the Material Database dialog box. These materials can not be modified or removed from the system. Contact IES Technical Support to suggest additional libraries of materials for IES to include in the database.

Custom Material Databases

Custom materials can be created in VisualPlate which are stored in the Custom Data Files. To add a custom material to the database in VisualPlate, click the Add Custom Material button in the Material Database dialog box. Use a General Material Type or choose the appropriate Material Type and edit the Defining Properties as needed. Custom materials can be used by VisualPlate, VisualAnalysis, ShapeBuilder, and other IES products (except Quick-products).

Limitations

All IES materials are assumed to be linear, isotropic and elastic. VisualPlate makes common use of orthotropic materials, like wood, but does so using isotropic properties. VisualPlate utilizes four primary properties: modulus of elasticity, Poisson's ratio, thermal coefficient of expansion, and weight density. Specific material types have additional properties that need to be defined (e.g. steel has yield stress, concrete has compressive strength, etc.).

Data Format

The database consists of XML data files. XML is a text based format which is commonly used for data-exchange and can be edited easily using a simple text editor (NotePad++ is recommended since it is freeware and offers XML syntax highlighting). Microsoft Word and similar programs are not recommend as they have a tenancy to corrupt the format.

Material Management

Look for instructions and examples, in the Customer\Materials folder. Database files can be copied to other computers where IES products are installed. Files that are no longer needed can be deleted. Files can be edited to change data or to remove shapes or shape categories. Make sure to backup the customized files.

2.10 Analysis

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

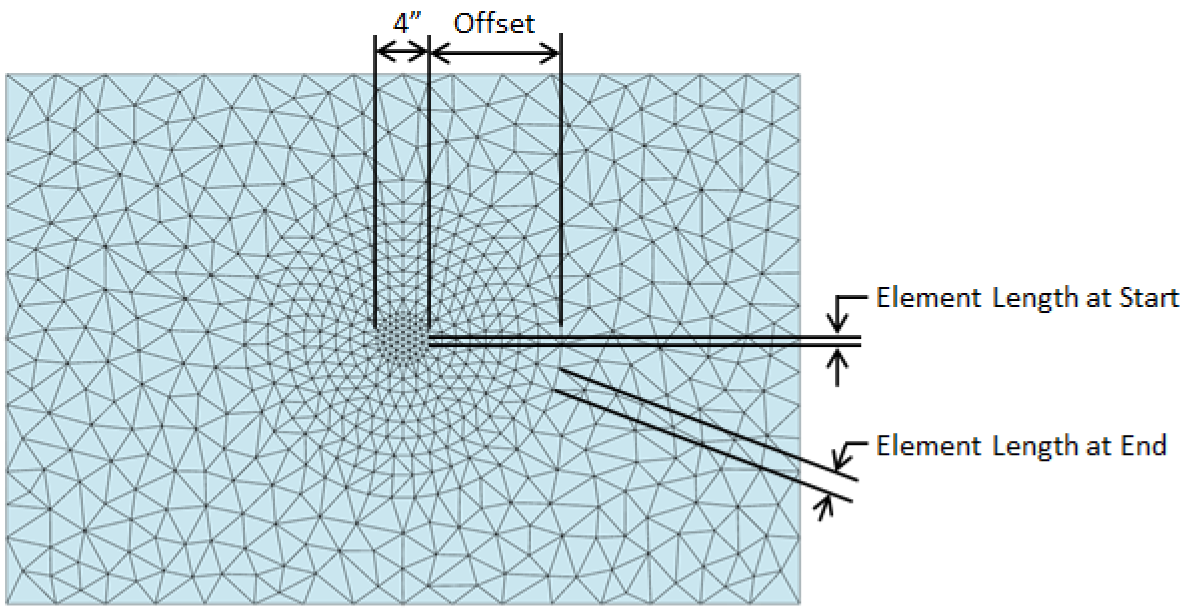
Mesh Settings

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 VisualPlate, the advanced finite element meshing options can be modified if needed. Note: The mesh settings at Load Points and Point 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 VisualPlate. The FEA 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 plate boundary). 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 VisualPlate, 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 Load Points and Point 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. load point locations that receive concentrated loads and point support locations where the support at a single node can create stress concentrations).

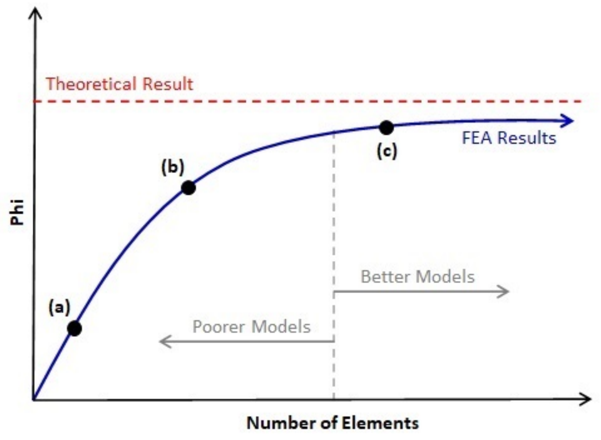


Load Point & Point Support Mesh Refinement

Mesh Refinement Procedure

1. Model the boundary and VisualPlate 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 boundary and/or refine the mesh at Load Points and Point 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.



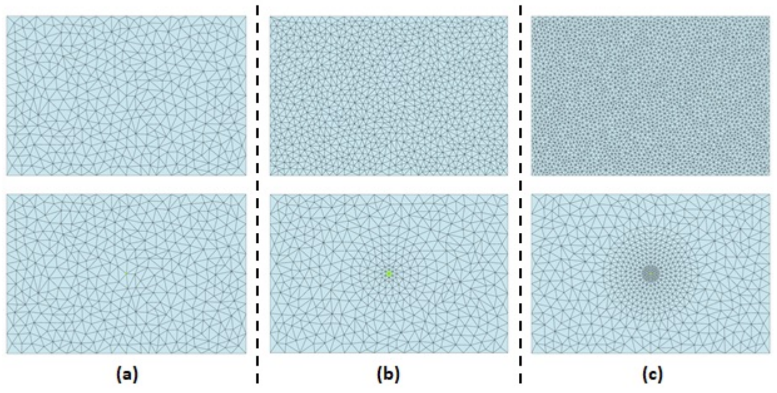


Plate Bending

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 VisualPlate. 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. VisualPlate 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.

References

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

3 Report

3.1 Reports

Reports in VisualPlate 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
- Member 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 VisualPlate. This feature is also available in other IES tools.

3.2 Tables

In VisualPlate, tables are used to report information in a clear and concise manner. The tables available for the report are listed in the **Project Manager | Tables** tab when the Report View is active. Tables fall into one of five categories (Project, Structure, Load, and Result) and will automatically appear or disappear depending on the items in the model (elements, loads, etc.) and the available analysis 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 Boundaries, Load Points, Point Supports, Line Supports, Line Stiffeners, 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, Point Loads, Line Loads, etc.
- **Result Tables** are used to document the analysis results for the elements in the model including Plate Forces and Displacements, Beam Forces and Displacements, Point and Line Support Results, etc.

Adding & Removing Tables

To add a table to the report, simply ***drag*** the table from the **Project Manager | 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 | Tables** 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 | Report Filter** 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 Boundaries, Point Supports, Load Points, Line Stiffeners, 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 Selected Table section in the **Project Manager | Report Filter** 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 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 | Report Filter** 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 VisualPlate. While Project Reports are saved in the .vpp project file, Report Styles are saved in the Custom Data Folder and can be used for multiple Projects. VisualPlate 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 .vpp 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 Custom Data Folder in an XML file and makes the style available for use in other VisualPlate 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 Custom 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 Member Graphs

Member Graphs display detailed diagrams (displacement, moment, shear, and torsion) for line stiffeners along their length. The results can be displayed for one single line stiffener or for a chain of line stiffeners for a single Result Case.

Create a Single or Multi Member Graph

To create a single member graph, simply select a line stiffener in the Model and Load or Analysis Results view and switch to the Member Graph view. A different line stiffener can be chosen from the dropdown in the **Project Manager | Graph Filter** tab. To create a multi member graph, simply select two or more connected line stiffeners that form a line and switch to the Member Graph view. If the selected line stiffeners 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 Member Graph

Member 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 Member Graph

Member 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 Member Graph

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

4 Integration

4.1 Integration with IES Tools

VisualAnalysis

VisualPlate can be used to create a VisualAnalysis project file (.vap) using the **File | Export a VA Project** feature. The .vap file can be used to look at the finite element model that was created behind the scenes in VisualPlate or to perform more advanced analyses (such as a dynamic analysis) which are not supported in VisualPlate. Note: VisualAnalysis is licensed separately by IES. Contact Sales for additional information regarding this product.

5 Script

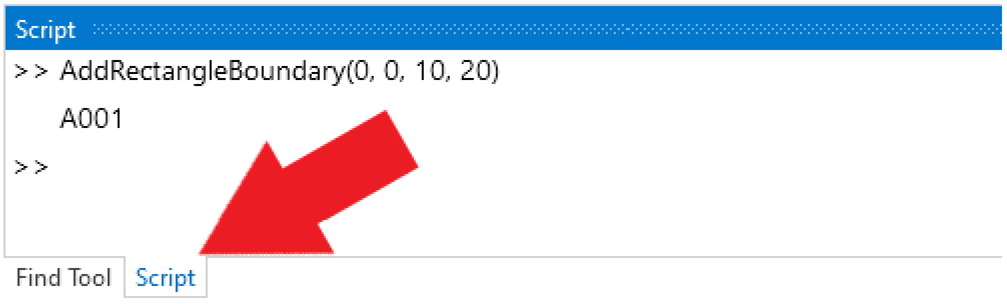
5.1 Script Overview

Introduction

The script feature in VisualPlate 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.

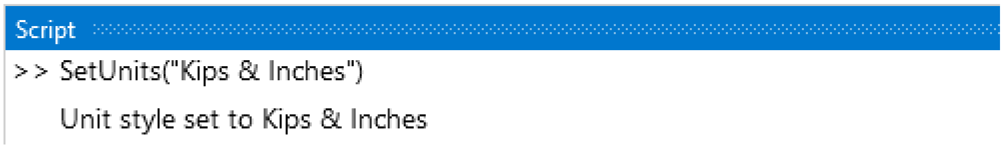
Command Line Basics

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.



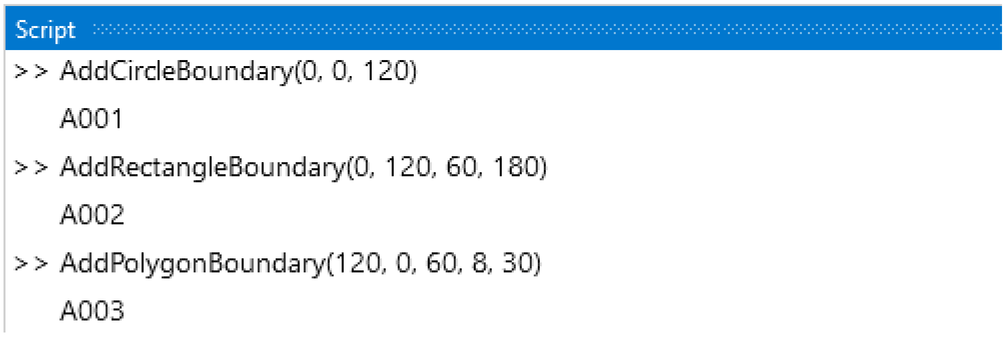
```
>> AddRectangleBoundary(0, 0, 10, 20)
   A001
>>
```

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



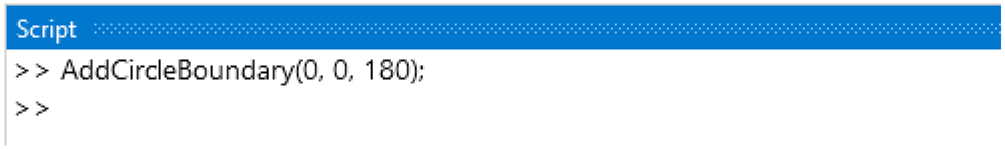
```
>> SetUnits("Kips & Inches")
   Unit style set to Kips & Inches
```

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



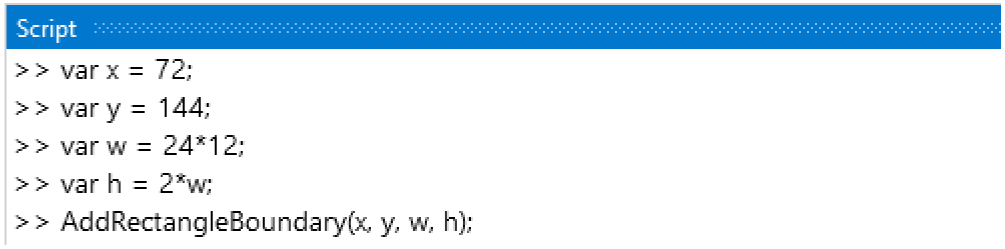
```
>> AddCircleBoundary(0, 0, 120)
   A001
>> AddRectangleBoundary(0, 120, 60, 180)
   A002
>> AddPolygonBoundary(120, 0, 60, 8, 30)
   A003
```

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



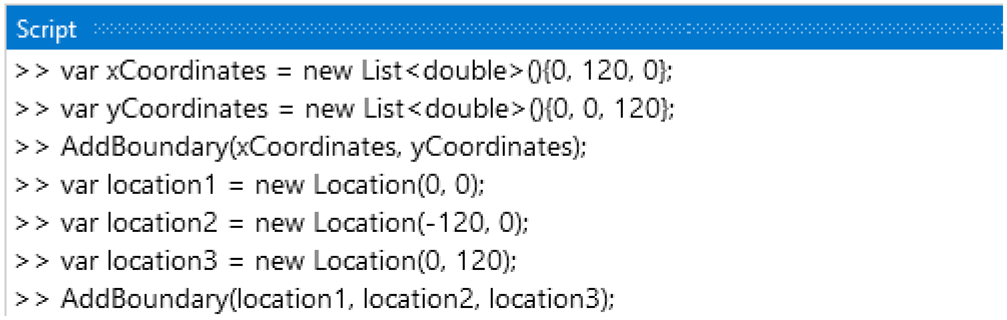
```
>> AddCircleBoundary(0, 0, 180);
>>
```

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 boundary (assuming the current unity style is Kips & Inches). Math is then used in the in the command line to define height of the boundary as two times the width. Note: When storing a value in a variable, the line must end with a semicolon.



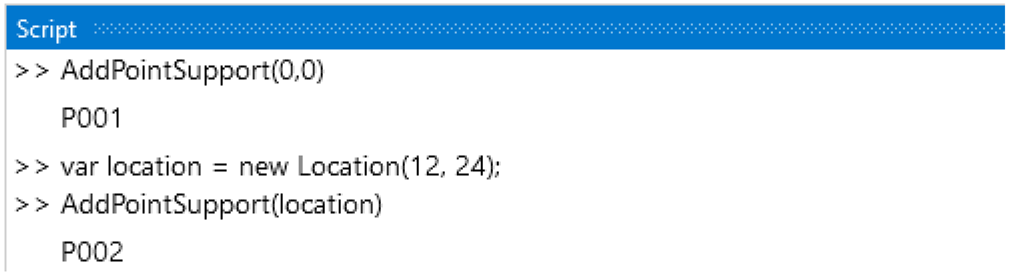
```
>> var x = 72;
>> var y = 144;
>> var w = 24*12;
>> var h = 2*w;
>> AddRectangleBoundary(x, y, w, h);
```

d. Boundaries 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.



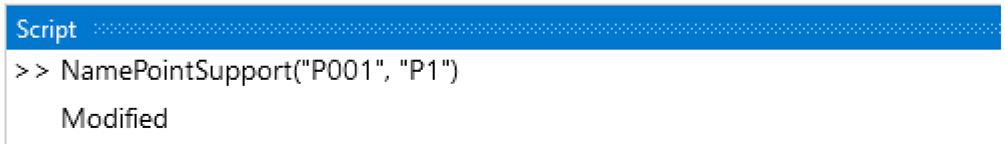
```
>> var xCoordinates = new List<double>(){0, 120, 0};
>> var yCoordinates = new List<double>(){0, 0, 120};
>> AddBoundary(xCoordinates, yCoordinates);
>> var location1 = new Location(0, 0);
>> var location2 = new Location(-120, 0);
>> var location3 = new Location(0, 120);
>> AddBoundary(location1,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

4. Add Point Supports
   a. Point Supports can be added to the project at specific coordinates or Locations.



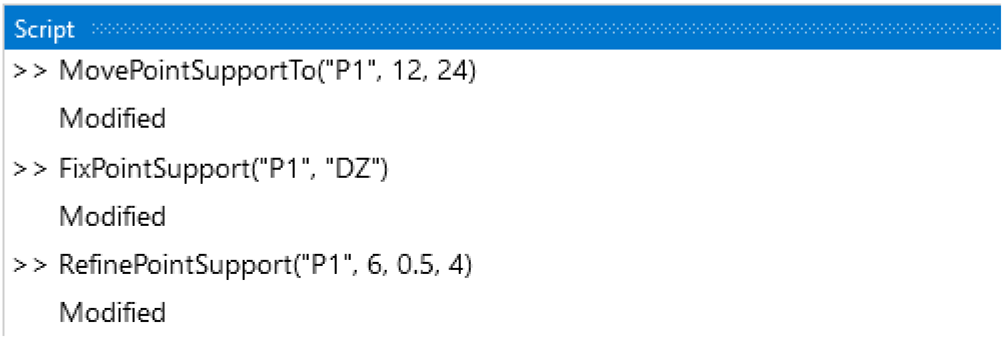
```
>> AddPointSupport(0,0)
   P001
>> var location = new Location(12, 24);
>> AddPointSupport(location)
   P002
```

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



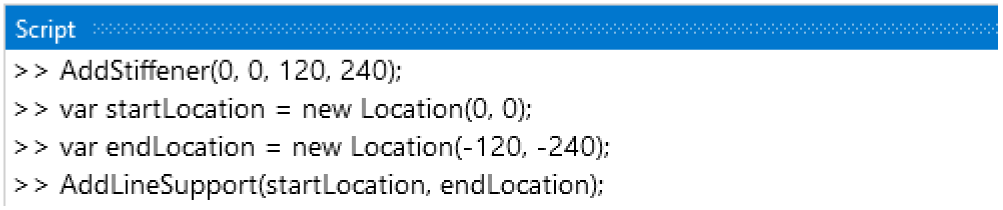
```
>> NamePointSupport("P001", "P1")
   Modified
```

   c. Once the Point Support is added, it will automatically be selected, and its properties can be modified in the Project Manager as usual. Alternatively, there are various commands to modify the Point Support parameters, see the Commands page for a complete list. Note: Item names must be enclosed in quotes.



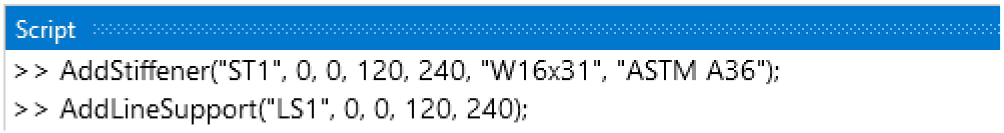
```
>> MovePointSupportTo("P1", 12, 24)
   Modified
>> FixPointSupport("P1", "DZ")
   Modified
>> RefinePointSupport("P1", 6, 0.5, 4)
   Modified
```

5. Add Line Stiffeners/Line Supports
   a. Line Stiffeners and Line Supports can be added to the project based on start and end coordinates or locations.



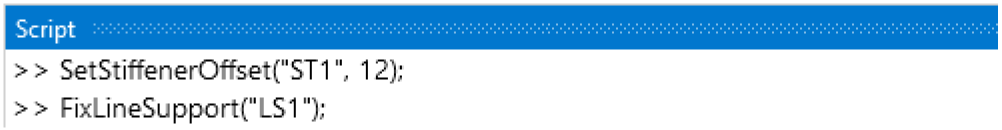
```
>> AddStiffener(0, 0, 120, 240);
>> var startLocation = new Location(0, 0);
>> var endLocation = new Location(-120, -240);
>> AddLineSupport(startLocation, endLocation);
```

   b. Various properties can be set when creating Line Stiffeners or Line Supports.



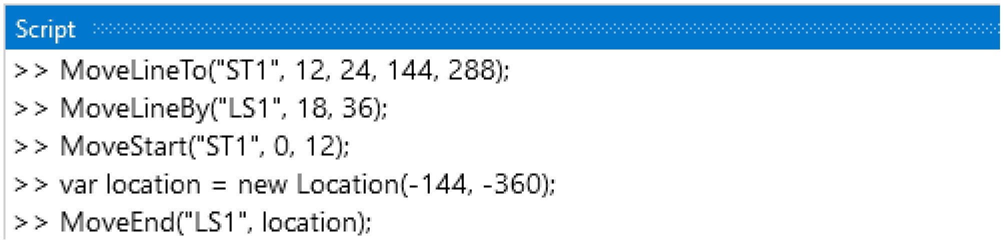
```
>> AddStiffener("ST1", 0, 0, 120, 240, "W16x31", "ASTM A36");
>> AddLineSupport("LS1", 0, 0, 120, 240);
```

   c. A variety of parameters can be modified for existing Line Stiffeners or Line Supports.



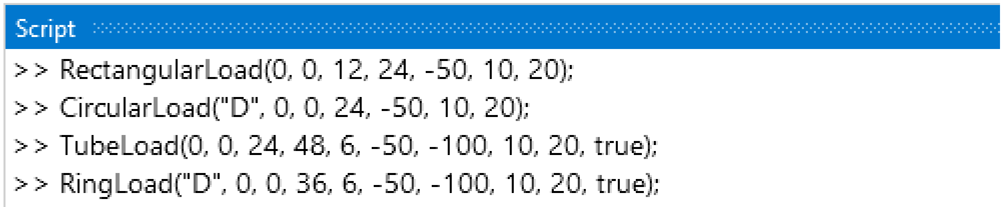
```
>> SetStiffenerOffset("ST1", 12);
>> FixLineSupport("LS1");
```

   d. Line Stiffeners and Line Supports can be moved using the command line.



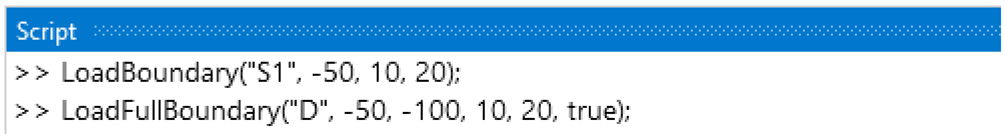
```
>> MoveLineTo("ST1", 12, 24, 144, 288);
>> MoveLineBy("LS1", 18, 36);
>> MoveStart("ST1", 0, 12);
>> var location = new Location(-144, -360);
>> MoveEnd("LS1", location);
```

6. 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.



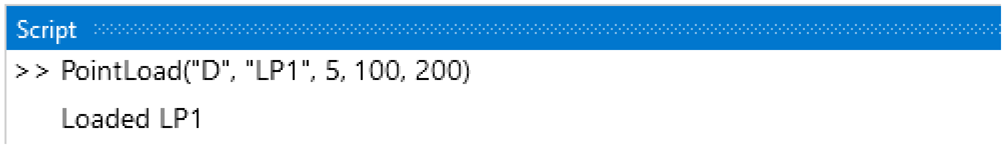
```
>> RectangularLoad(0, 0, 12, 24, -50, 10, 20);
>> CircularLoad("D", 0, 0, 24, -50, 10, 20);
>> TubeLoad(0, 0, 24, 48, 6, -50, -100, 10, 20, true);
>> RingLoad("D", 0, 0, 36, 6, -50, -100, 10, 20, true);
```

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



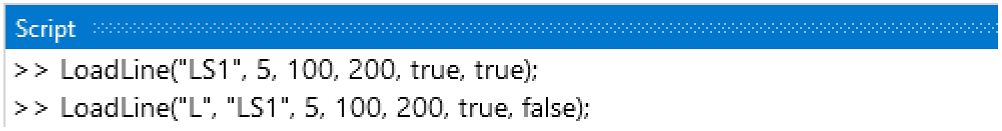
```
Script
>> LoadBoundary("S1", -50, 10, 20);
>> LoadFullBoundary("D", -50, -100, 10, 20, true);
```

c. A variety of options are available to apply loads to Load Points using the command line as outlined on the Commands page. In the example below, Load Point LP1 is loaded in the D service case with a 5 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).



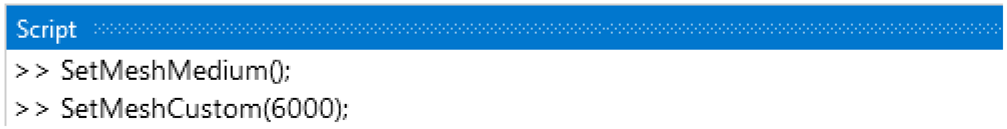
```
Script
>> PointLoad("D", "LP1", 5, 100, 200)
   Loaded LP1
```

d. Line Stiffeners are loaded using the LoadLine command which can take various inputs.



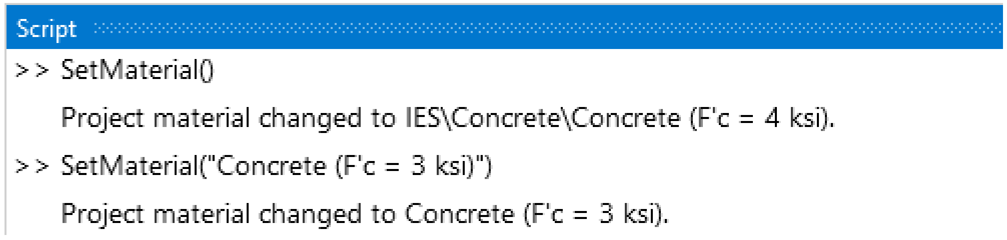
```
Script
>> LoadLine("LS1", 5, 100, 200, true, true);
>> LoadLine("L", "LS1", 5, 100, 200, true, false);
```

7. Analysis Settings
   a. The command line can be used to set the mesh to a standard value (Coarse, Medium, or Fine) or to a User Defined value.



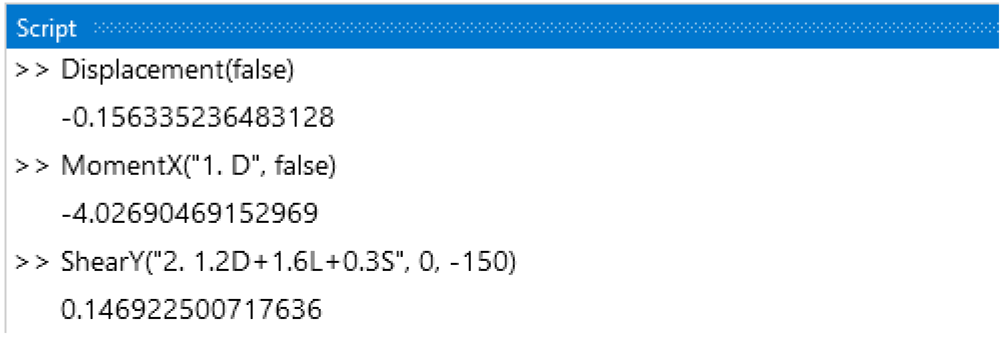
```
Script
>> SetMeshMedium();
>> SetMeshCustom(6000);
```

   b. The SetMaterial() command can be used to set the material for the project. This command launches the material dialog box so that a material can be chosen. Alternatively, the name of the material can be used in the command to directly set the material.



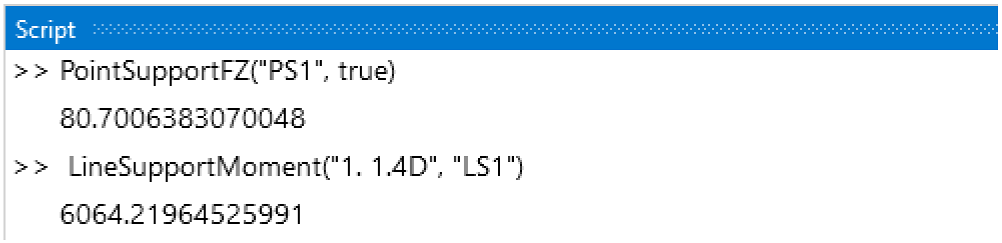
```
Script
>> SetMaterial()
   Project material changed to IES\Concrete\Concrete (F'c = 4 ksi).
>> SetMaterial("Concrete (F'c = 3 ksi)")
   Project material changed to Concrete (F'c = 3 ksi).
```

8. Obtain Results
   a. The maximum or minimum displacement, shear or moment 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
>> Displacement(false)
   -0.156335236483128
>> MomentX("1. D", false)
   -4.02690469152969
>> ShearY("2. 1.2D+1.6L+0.3S", 0, -150)
   0.146922500717636
```

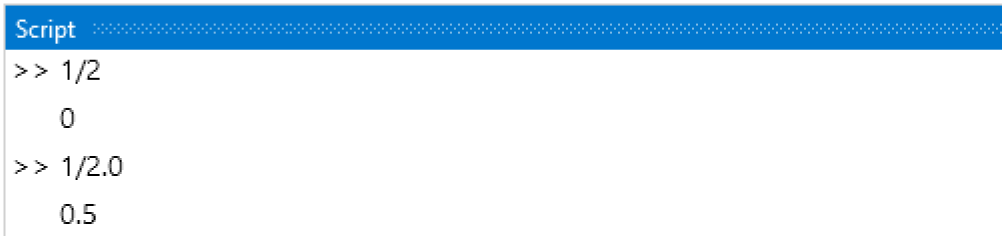
   b. The maximum or minimum Point Support or Line Support force or moment across all result cases can be obtained using command line. Also, the command line can output the force at a specified Point Support or Line Support Point for a defined result case.



Script

```
>> PointSupportFZ("PS1", true)
   80.7006383070048
>>  LineSupportMoment("1. 1.4D", "LS1")
   6064.21964525991
```

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.

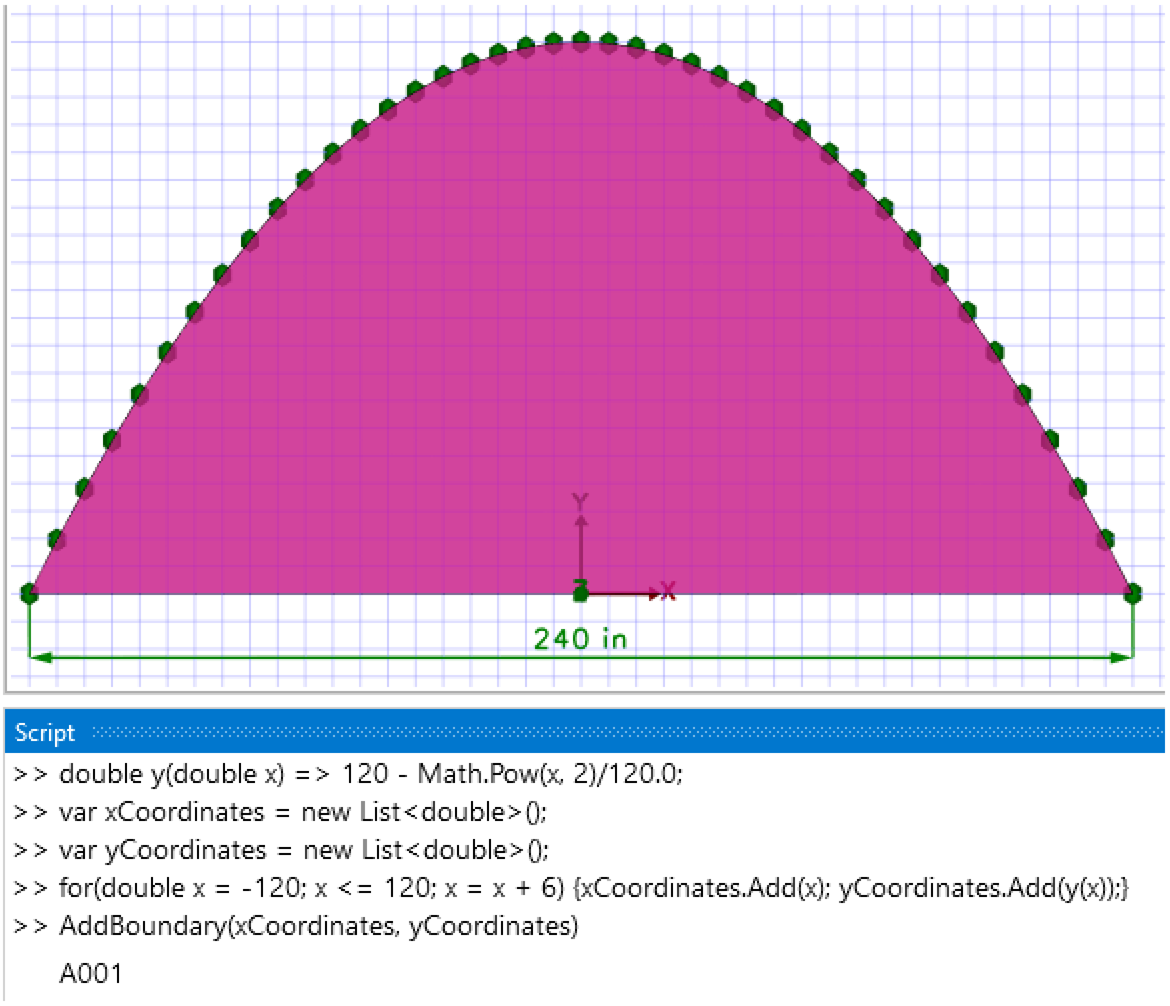


Script

```
>> 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 boundary to lists which are then used to define the boundary. 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.



```
>> 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));}
>> AddBoundary(xCoordinates, yCoordinates)
   A001
```

5.2 Commands

Script Command Categories

- User Interface
- General
- Vertices
- Boundaries
- Point Supports
- Line Supports
- Line Stiffeners
- Move Line Stiffener/Line Support
- Service Cases
- Load Points
- Boundary Loads
- Point Loads
- Line Stiffener Loads
- Delete Loads
- Load Combinations
- Analysis Settings
- Status
- Pipeline
- Result Cases

- Plate Results
- Boundary Results
- Point Support Results
- Line Support Results
- Report

	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("AddBoundary")	Launches the Help File and navigates to the AddBoundary 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
	SetDisplayPrecision(DecimalPlaces)	Sets the number of decimal places displayed	SetDisplayPrecision(4)	Sets the precision to 4 decimal places
	Select(names[])	Selects a specified item(s) (e.g. boundary, vertex, etc.) in the model	Select("B1", "V1")	Selects boundary B1 and vertex V1 in the model
	Delete()	Deletes the selected model object(s) and/or load(s)	Select boundary B1, element Bm1, and vertex V1 then enter Delete()	Deletes boundary B1, element Bm1, and vertex V1 in the model
	Delete(names[])	Deletes a specified item(s) (e.g. boundaries, load points, point supports, etc.) in the model	Delete("B1", "LP1", "PS1")	Deletes boundary B1, load point LP1, and point support PS1 in the model
	DeleteAll()	Deletes everything in the model		
	Zoom(name)	Zooms to a specified item (e.g. boundary, load point, point support, etc.) in the	Zoom("V1")	Zooms into vertex V1 in the model

		model		
	Print(List<names>)	Prints the list of items in a comma-delimited format	Print(Boundaries())	Prints the list of boundaries 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
	PickMaterial()	Opens the Material Database dialog box	AddStiffener("ST1", 0, 0, 120, 240, "W16x31", PickMaterial())	Opens the Material Database dialog box to define a material for line stiffener ST1 that is created between {0, 0} and {120, 240}
	PickDatabaseSection()	Opens the Shape Database dialog box	AddStiffener("ST1", 0, 0, 120, 240, PickDatabaseSection(), "ASTM A36")	Opens the Shape Database dialog box to define a database section for line stiffener ST1 that is created between {0, 0} and {120, 240}
	PickParametricSection()	Opens the Parametric Shape Dimensions dialog box	AddStiffener("ST1", 0, 0, 120, 240, PickParametricSection(), "ASTM A36")	Opens the Parametric Shape Dimensions dialog box to define a parametric section for line stiffener ST1 that is created between {0, 0} and {120, 240}
	SetTitle()	Sets the Title for the project	SetTitle("Elevated Slab Design")	Sets the Title for the project to "Elevated Slab Design"
	SetBillingReference()	Sets the Billing Reference for the project	SetBillingReference("ABC Architects")	Sets the Billing Reference for the project to "ABC Architects"
	SetProjectNotes()	Adds a project note	SetProjectNotes("Model complete.")	Adds note "Model complete." to the project.
	SetNodalTolerance()	Sets the Nodal Tolerance	SetNodalTolerance(0.075)	Sets the Nodal Tolerance to 0.075
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	MoveTo("V1", 0, 0)	Moves vertex V1 to the origin

		location		

	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 the specified vertex	X("V1")	Returns the global X-coordinate of vertex V1
	Y(name)	Returns the global Y-coordinate of the specified vertex	Y("V1")	Returns the global Y-coordinate of vertex V1
Boundaries (BACK TO TOP)	Boundaries()	Returns a list of all the boundaries in the project	Print(Boundaries())	Prints the list of all the boundaries in the project in a comma-delimited format

	AddBoundary(List<X>, List<Y>)	Adds a boundary 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};
			AddBoundary(X, Y)
				Adds a boundary defined by coordinate lists X and Y

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

	AddRectangleBoundary(centerX, centerY, width, height)	Adds a rectangular boundary of specified width and height at a defined location	AddRectangleBoundary(1, 2, 5, 15)	Adds a 5 wide by 15 tall rectangular boundary centered at {1, 2}
	AddCircleBoundary(centerX, centerY, radius)	Adds a circular boundary of specified radius at a defined location	AddCircleBoundary(1, 2, 5)	Adds a circular boundary of radius 5 centered at {1, 2}
	AddPolygonBoundary(centerX, centerX, radius, numberSides, angle)	Adds a polygon boundary of specified radius at a defined location	AddPolygonBoundary(1, 2, 5, 6, 45)	Adds a 6 sided polygon boundary of radius 5 centered at {1, 2} with a rotation angle of 45
	ToggleHole(boundary)	Toggles the Hole? parameter for the specified boundary on or off	ToggleHole("B1")	Toggles the Hole? parameter on or off for boundary B1
	Thickness(boundary, thickness)	Sets the thickness to a specified value for the defined boundary	Thickness("B1", 12)	Sets the thickness for boundary B1 to 12
	ToggleSelfWeight(boundary)	Toggles the Add Self Weight parameter	ToggleSelfWeight("B1")	Toggles the Add Self Weight

		for the specified boundary on or off		parameter on or off for boundary B1
	Height(boundary, height)	Modifies the height of a rectangular or circular boundary	Height("B1", 120)	Changes the height of boundary B1 to 120
	Width(boundary, width)	Modifies the width of a rectangular or circular boundary	Width("B1", 120)	Changes the width of boundary B1 to 120
	Theta(boundary, theta)	Modifies the theta of a rectangular or polygon boundary	Theta("B1", 45)	Changes theta of boundary B1 to 45
	Radius(boundary, radius)	Modifies the radius of a circular or polygon boundary	Radius("B1", 120)	Changes the radius of boundary B1 to 120
	SideCount(boundary, sideCount)	Modifies the number of sides for a polygon boundary	SideCount("B1", 6)	Changes the number of sides for boundary B1 to 6
	SideLength(boundary, length)	Modifies the side length for a polygon boundary	SideLength("B1", 120)	Changes the side length for boundary B1 to 120
	MoveCenterTo(boundary, centerX, centerY)	Moves the center of the a circular, rectangular, or polygon boundary to a defined location	MoveCenterTo("B1", 120, 240)	Moves the center of boundary B1 to {120, 240}

	MoveCenterTo(boundary, location)	Moves the center of the a circular, rectangular, or polygon boundary to a defined location	var location1 = new Location(120, 240);
			MoveCenterTo("B1", location1)
				Moves the center of boundary B1 to location1

	MoveCenterBy(boundary, distanceX, distanceY)	Moves the center of the a circular, rectangular, or polygon slab by a specified amount	MoveCenterBy("F1", 120, 240)	Moves the center of slab F1 by 120 in the X-direction and 240 in the Y-direction
Point Supports (BACK TO TOP)	PointSupports()	Returns a list of all the point supports in the project	Print(PointSupports())	Prints the list of all the point supports in the project in a comma-delimited format
	AddPointSupport(X, Y)	Adds a default point support at a specified coordinate	AddPointSupport(0, 0)	Adds a default point support at the origin

	AddPointSupport(location)	Adds a default point support at a specified location	var location1 = new Location(12, 24);
			AddPointSupport(location1)
				Adds a default point support at location1

	NamePointSupport(currentName, newName)	Renames a point support	NamePointSupport("PS1", "PS2")	Changes the name of point support PS1 to PS2
	FixPointSupport(support, dof)	Sets the specified degree of freedom to fixed for the defined	FixPointSupport("PS1", "DZ")	Sets Force-Z to fixed for point support PS1

		point support		
	FreePointSupport(support, dof)	Sets the specified degree of freedom to free for the defined point support	FreePointSupport("PS1", "RX")	Sets Moment-X to free for point support PS1
	CustomPointSupport(support, dof, stiffness)	Sets the specified degree of freedom to a spring support with specified stiffness for the defined point support	CustomPointSupport("PS1", "RY", 100)	Sets Moment-Y to Specified-K for point support PS1 with a Stiffness-Ry = 100
	RefinePointSupport(support, offset, startLength, endLength)	Sets the Refine parameter for the specified point support to true and defines the mesh refinement parameters	RefinePointSupport("PS1", 6, 0.5, 4)	Sets the Refine parameter for point support PS1 to true and set the offset to 6, the element length at start to 0.5, and the element length at end to 4
	UnRefinePointSupport(support)	Sets the Refine parameter for the specified point support to false	UnRefinePointSupport("PS1")	Sets the Refine parameter for point support PS1 to false
	MovePointSupportTo(support, X, Y)	Moves specified point support to defined coordinates	MovePointSupportTo("PS1", 12, 24)	Moves point support PS1 to {12, 24}

	MovePointSupportTo(support, location)	Moves specified point support to defined location	var location1 = new Location(12, 24);
			MovePointSupportTo("PS1", location1)
				Moves point support PS1 to location1

	MovePointSupportBy(support, distanceX, distanceY)	Moves specified point support by a specified distance in each global direction	MovePointSupportBy("PS1", 12, 24)	Moves point support PS1 by 12 in the X-direction and 24 in the Y-direction
Line Supports (BACK TO TOP)	LineSupports()	Returns a list of all the line supports in the project	Print(LineSupports())	Prints the list of all the line supports in the project in a comma-delimited format
	AddLineSupport(startX, startY, endX, endY)	Adds a line support between start coordinates and end coordinates	AddLineSupport(0, 0, 120, 240)	Adds a line support between {0, 0} and {120, 240}

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

	AddLineSupport(name, startX, startY, endX, endY)	Adds a line support between start coordinates and end coordinates with a defined name	AddLineSupport("LS1", 0, 0, 120, 240)	Adds a line support named LS1 between {0, 0} and {120, 240}

	AddLineSupport(name, startLocation, endLocation)	Adds a line support between a start location and an end location with a	var location1 = new Location(0, 0);
			var location2 = new Location(120, 240);
			AddLineSupport("LS1", location1, location2)
				Adds a line support named LS1 between location1 and

		defined name		location2
	FixLineSupport(support)	Sets the moment fixity at the support to fixed	FixLineSupport("LS1")	Sets the moment fixity for line support LS1 to fixed
	PinLineSupport(support)	Sets the moment fixity at the support to pinned	PinLineSupport("LS1")	Sets the moment fixity for line support LS1 to pinned
	CustomLineSupport(support, stiffness)	Sets the moment fixity at the support to a spring with a specified stiffness	CustomLineSupport("LS1", 100)	Sets the moment fixity for line support LS1 to a spring of stiffness 100
Line Stiffeners (BACK TO TOP)	Stiffeners()	Returns a list of all the line stiffeners in the project	Print(Stiffeners())	Prints the list of all the line stiffeners in the project in a comma-delimited format
	AddStiffener(startX, startY, endX, endY)	Adds a line stiffener between start coordinates and end coordinates	AddStiffener(0, 0, 120, 240)	Adds a line stiffener between {0, 0} and {120, 240}
	AddStiffener(name, startX, startY, endX, endY, section, material)	Adds a line stiffener between start coordinates and end coordinates with a defined name, section, and material	AddStiffener("ST1", 0, 0, 120, 240, "W16x31", "ASTM A36")	Adds a line stiffener named ST1 between {0, 0} and {120, 240} that has a W16x31 section with A36 material
	AddStiffener(name, startX, startY, endX, endY, section, material, offsetZ, rotation, includeSelfWeight)	Adds a line stiffener between start coordinates and end coordinates with a defined name, section, material, offset, rotation, and with the self weight included or excluded	AddStiffener("ST1", 0, 0, 120, 240, "W16x31", "ASTM A36", 12, 90, true)	Adds a line stiffener named ST1 between {0, 0} and {120, 240} that has a W16x31 section with A36 material that is offset 12 in the Z-direction, rotated 90, and includes the self weight

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

	AddStiffener(name, startLocation, endLocation, section, material)	Adds a line stiffener between a start location and an end location with a defined name, section, and material	var location1 = new Location(0, 0);
			var location2 = new Location(120, 240);
			AddStiffener("ST1", location1, location2, "W16x31", "ASTM A36")
				Adds a line stiffener named ST1 between location1 and location2 that has a W16x31 section with A36 material

	AddStiffener(name, startLocation,	Adds a line stiffener	var location1 = new Location(0, 0);	Adds a line

	endLocation, section, material, offsetZ, rotation, includeSelfWeight)	between a start location and an end location with a defined name, section, material, offset, rotation, and with the self weight included or excluded	var location2 = new Location(120, 240);
			AddStiffener("ST1", location1, location2, "W16x31", "ASTM A36", 12, 90, true)
				stiffener named ST1 between location1 and location2 that has a W16x31 section with A36 material that is offset 12 in the Z-direction, rotated 90, and includes the self weight

	SetStiffenerOffset(stiffener, offsetZ)	Sets the offset in the Z-direction for the specified line stiffener	SetStiffenerOffset("ST1", 12)	Sets the offset in the Z-direction for ST1 to 12
	SetStiffenerRotation(stiffener, angle)	Sets the rotation for the specified line stiffener	SetStiffenerRotation("ST1", 90)	Sets the rotation for ST1 to 90
	SetStiffenerSelfWt(stiffener, include)	Includes or excludes the self weight for the specified line stiffener	SetStiffenerSelfWt("ST1", true)	Includes the self weight for ST1
Move Line Stiffener/Line Support (BACK TO TOP)	MoveLineTo(line, startX, startY, endX, endY)	Moves specified line stiffener or line support to a defined start coordinate and end coordinate	MoveLineTo("L1", 0, 0, 120, 240)	Moves line stiffener or line support L1 to span between coordinates {0, 0} and {120, 240}

	MoveLineTo(line, startLocation, endLocation)	Moves specified line stiffener or line support 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 line stiffener or line support L1 to span between location1 and locaiton2

	MoveLineBy(line, distanceX, distanceY)	Moves specified line stiffener or line support by specified amounts in the X-direction and the Y-direction	MoveLineBy("L1", 12, 24)	Moves line stiffener or line support L1 by 12 in the X-direction and 24 in the Y-direction
	MoveStart(line, X, Y)	Moves the start point of a specified line stiffener or line support to a defined coordinate	MoveStart("L1", 6, 12)	Moves the start coordinate of line stiffener or line support L1 to {6, 12}

	MoveStart(line, startLocation)	Moves the start point of a specified line stiffener or line support to a defined location	var location1 = new Location(0, 0);
			MoveStart("L1", location1)
				Moves the start coordinate of line stiffener or line support L1 to locatoin1

	MoveEnd(line, X, Y)	Moves the end point of a specified line stiffener or line support to a defined coordinate	MoveEnd("L1", 6, 12)	Moves the end coordinate of line stiffener or line support L1 to {6, 12}

	MoveEnd(line, endLocation)	Moves the end point of a specified line stiffener or line support to a defined location	var location1 = new Location(120, 120);
			MoveEnd("L1", location1)
				Moves the end coordinate of line stiffener or line support L1 to location1

Service Cases [(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
Load Points [(BACK TO TOP)]	LoadPoints()	Returns a list of all the load points in the project	Print(LoadPoints())	Prints the list of all the load points in the project in a comma-delimited format
	AddLoadPoint(X, Y)	Adds a default load point at a specified coordinate	AddLoadPoint(0, 0)	Adds a default load point at the origin

	AddLoadPoint(location)	Adds a default load point at a specified location	var location1 = new Location(12, 24);
			AddLoadPoint(location1)
				Adds a default load point at location1

	NameLoadPoint(currentName, newName)	Renames a load point	NameLoadPoint("LP1", "LP2")	Changes the name of point support PS1 to PS2
	RefineLoadPoint(point, offset, startLength, endLength)	Sets the Refine parameter for the specified load point to true and defines the mesh refinement parameters	RefineLoadPoint("LP1", 6, 0.5, 4)	Sets the Refine parameter for laod point LP1 to true and set the offset to 6, the element length at start to 0.5, and the element length at end to 4
	UnRefineLoadPoint(support)	Sets the Refine parameter for the specified load point to false	UnRefineLoadPoint("LP1")	Sets the Refine parameter for load point LP1 to false
	MoveLoadPointTo(point, X, Y)	Moves specified load point to defined coordinates	MoveLoadPointTo("LP1", 12, 24)	Moves load point LP1 to {12, 24}

	MoveLoadPointTo(point, location)	Moves specified load point to defined location	var location1 = new Location(12, 24);
			MoveLoadPointTo("LP1", location1)
				Moves load point LP1 to location1

	MoveLoadPointBy(point, distanceX, distanceY)	Moves specified load point by a specified distance in	MoveLoadPointBy("LP1", 12, 24)	Moves load point LP1 by 12 in the X-

		each global direction		direction and 24 in the Y-direction
Boundary 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 at {X, Y} with a specified uniform pressure with MX and MY overturning loads in the defined service case	RectangularLoad("D", 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 "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 radius centered at {X, Y} with a specified	CircularLoad("D", 0, 0, 4, -50, 10, 20)	Applies a circular load with a 4 radius centered at {0, 0} with a -

	uniform pressure with MX and MY overturning loads in the defined service case		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, 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 defined service case	TubeLoad("D", 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 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,	Applies a widthX by widthY tube load	TubeLoad("D", 0, 0, 2, 4, 0.5, -50, -100, 10, 20, true)	Applies a 2 by 4 tube load

	startPressure, endPressure, MX, MY, linearX)	centered at {X, Y} with a specified linear pressure with MX and MY overturning loads in the defined service case		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 thickness centered at {X, Y} with a specified uniform pressure with MX and MY overturning loads in the defined service case	RingLoad("D", 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 "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
	LoadBoundary(uniformPressure, MX, MY)	Applies a uniform pressure with MX and MY overturning loads to the selected boundary in the	LoadBoundary(-50, 10, 20)	Applies a -50 uniform pressure with 10 and 20 overturning loads to the

		current service case		selected boundary in the current service case
	LoadBoundary(slab, uniformPressure, MX, MY)	Applies a uniform pressure with MX and MY overturning loads to the specified boundary in the current service case	LoadBoundary("S1", -50, 10, 20)	Applies a -50 uniform pressure with 10 and 20 overturning loads to boundary S1 in the current service case
	LoadBoundary(serviceCase, boundary, uniformPressure, MX, MY)	Applies a uniform pressure with MX and MY overturning loads to the specified boundary in the defined service case	LoadBoundary("D", "S1", -50, 10, 20)	Applies a -50 uniform pressure with 10 and 20 overturning loads to boundary S1 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
Point Loads (BACK TO TOP)	PointLoad(FZ, MX, MY)	Applies specified force and moments to the selected load point in the current service case	PointLoad(5, 100, 200)	Applies force and moments to the selected load point in the current service case
	PointLoad(loadPoint, FZ, MX, MY)	Applies specified force and moments to the defined load	PointLoad("LP1", 5, 100, 200)	Applies forces and moments to load point "LP1"

		point in the current service case		in the current service case
	PointLoad(serviceCase, loadPoint, FZ, MX, MY)	Applies specified force and moments to the defined load point in the specified service case	PointLoad("D", "LP1", 5, 100, 200)	Applies forces and moments to load point "LP1" in the D service case
Line Stiffener Loads (BACK TO TOP)	LoadLine(FZ, MX, MY, distributed = false, local = false)	Applies specified force and moments to the selected line stiffener in the current service case as a resultant or distributed load in the global or local direction	LoadLine(5, 100, 200)	Applies force and moments to the selected line stiffener in the current service case. The loads are applied as resultants in the global direction since "distributed" and "local" are false by default.
	LoadLine(beam, FZ, MX, MY, distributed = false, local = false)	Applies specified force and moments to the defined line stiffener in the current service case as a resultant or distributed load in the global or local direction	LoadLine("LS1", 5, 100, 200, true, true)	Applies force and moments to line stiffener LS1 in the current service case. The loads are applied as distributed loads in the local direction.
	LoadLine(serviceCase, beam, FZ, MX, MY, distributed = false, local = false)	Applies specified forces and moments to the defined line stiffener in the defined service case as a resultant or distributed load in the global or local direction	LoadLine("D", "LS1", 5, 100, 200, true, false)	Applies force and moments in the D service case to the line stiffener LS1 in the current service case. The loads are applied as distributed loads in the global direction.
Delete Loads (BACK TO TOP)	ClearLoads()	Deletes all of the loads in the active service case		
	ClearLoads(case)	Deletes all of the loads in the specified service case	ClearLoads("L")	Deletes all of the loads in the L service case
	DeleteLoad(object)	Deletes the loads on a specified object in the active service case	DeleteLoad("ST1")	Deletes the wall loads on line stiffener ST1 in the active service case
	DeleteLoad(case, object)	Deletes the loads on a specified object in the specified service case	DeleteLoad("L", "ST1")	Deletes the wall loads on line stiffener ST1 in the L service case
Load Combinations (BACK TO TOP)	AddCustomCombo(name, comboType, (factor, case)[])	Adds a custom load combination to the project	AddCustomCombo("D + 0.75L", "Allowable (ASD)", (1.0, "D"), (0.75, "L"))	Adds custom D + 0.75L Allowable (ASD) load combination

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
	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 material that is used for the slabs and beams in the project	SetMaterial("Concrete (F'c = 3.5 ksi)")	Sets the concrete material that is used for the slabs and beams in the project to Concrete (F'c = 3.5 ksi)
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 2 = Ok	Status("Analysis")	Returns the Project Status of the Analysis
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
	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

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
	ServiceResults()	Returns a list of all the service result cases in the project	Print(ServiceResults())	Prints the list of all the service result cases in the project in a comma-delimited format
	StrengthResults()	Returns a list of all the strength result cases in the project	Print(StrengthResults())	Prints the list of all the strength result cases in the project in a comma-delimited format
	NoDesignResults()	Returns a list of all the no design result cases in the project	Print(NoDesignResults())	Prints the list of all the no design result cases in the project in a comma-delimited format
	SetVisibleRC(resultCase)	Sets the visible result case in the analysis results	SetVisibleRC("2. D+L")	Sets the visible result case in the analysis results to 2. D+L
Plate Results (BACK TO TOP)	Displacement(max)	Returns the maximum or minimum displacement in the global Z-direction across all result cases	Displacement(false)	Returns the minimum displacement in the global 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	ShearX("1. D", true)	Returns the maximum shear force that acts on the global X-face of the plate elements (VX) for the defined

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

		maximum or minimum bending moment that acts on the global Y-face of the plate elements (MY) across all result cases		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("2. D+L", 6, 12)	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 moment (MXY) for the defined result case	MomentXY("1. D", false)	Returns the minimum twisting moment (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
Boundary Results (BACK TO TOP)	BoundaryDisplacement(boundary, max)	Returns the maximum or minimum displacement in the global Z-direction across all result cases for the specified Boundary	BoundaryDisplacement("B1", false)	Returns the minimum displacement in the global Z-direction across all result cases for boundary B1
	BoundaryDisplacement(boundary, result, max)	Returns the maximum or minimum displacement in the global Z-direction for the defined result case for the specified boundary	BoundaryDisplacement("B1", "1. D", true)	Returns the maximum displacement in the in the global Z-direction for the "1. D" result case for boundary B1
	BoundaryShearX(boundary, max)	Returns the maximum or	BoundaryShearX("B1", false)	Returns the minimum shear

	minimum shear force that acts on the global X-face of the plate elements (VX) across all result cases for the specified boundary		force that acts on the global X-face of the plate elements (VX) across all result cases for boundary B1
BoundaryShearX(boundary, 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 for the specified boundary	BoundaryShearX("B1", "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 for boundary B1
BoundaryShearY(boundary, max)	Returns the maximum or minimum shear force that acts on the global Y-face of the plate elements (VY) across all result cases for the specified boundary	BoundaryShearY("B1", true)	Returns the maximum shear force that acts on the global Y-face of the plate elements (VY) across all result cases for boundary B1
BoundaryShearY(boundary, 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 for the specified boundary	BoundaryShearY("B1", "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 for boundary B1
BoundaryMomentX(boundary, max)	Returns the maximum or minimum bending moment that acts on the global X-face of the plate elements (MX) across all result cases for the specified boundary	BoundaryMomentX("B1", true)	Returns the maximum bending moment that acts on the global X-face of the plate elements (MX) across all result cases for boundary B1
BoundaryMomentX(boundary, 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 for the specified boundary	BoundaryMomentX("B1", "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 for boundary b1
BoundaryMomentY(boundary, max)	Returns the maximum or minimum bending moment that acts on the global Y-face of the plate elements (MY) across all result cases for the	BoundaryMomentY("B1", false)	Returns the minimum bending moment that acts on the global Y-face of the plate elements (MY)

		specified boundary		across all result cases for boundary B1
	BoundaryMomentY(boundary, 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 for the specified boundary	BoundaryMomentY("B1", "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 for boundary B1
	BoundaryMomentXY(boundary, max)	Returns the maximum or minimum twisting moment (MXY) across all result cases for the specified boundary	BoundaryMomentXY("B1", true)	Returns the maximum twisting moment (MXY) across all result cases for boundary B1
	BoundaryMomentXY(boundary, result, max)	Returns the maximum or minimum twisting moment (MXY) for the defined result case for the specified boundary	BoundaryMomentXY("B1", "1. D", false)	Returns the minimum twisting moment (MXY) for the ""1. D"" result case for boundary B1
Point Support Results (BACK TO TOP)	PointSupportFZ(result, support)	Returns Force Z for the specified support for the specified result case	PointSupportFZ("1. 1.4D", "PS1")	Returns Force Z for point support PS1 for result case 1. 1.4D
	PointSupportFZ(support, wantMax)	Returns the maximum or minimum Force Z for the specified point support	PointSupportFZ("PS1", true)	Returns the maximum Force Z for point support PS1 across all result cases
	PointSupportMX(result, support)	Returns Moment X for the specified point support for the specified result case	PointSupportMX("1. 1.4D", "PS1")	Returns Moment X for point support PS1 for result case 1. 1.4D
	PointSupportMX(support, wantMax)	Returns the maximum or minimum Moment X for the specified point support	PointSupportMX("PS1", true)	Returns the maximum Moment X for point support PS1 across all result cases
	PointSupportMY(result, support)	Returns Moment Y for the specified point support for the specified result case	PointSupportMY("1. 1.4D", "PS1")	Returns Moment Y for point support PS1 for result case 1. 1.4D
	PointSupportMY(support, wantMax)	Returns the maximum or minimum Moment Y for the specified point support	PointSupportMY("PS1", true)	Returns the maximum Moment Y for point support PS1 across all result cases
Line Support	LineSupportForce(result, support)	Returns the force for	LineSupportForce("1. 1.4D", "LS1")	Returns the

Results (BACK TO TOP)		the specified line support for the specified result case		force for line support LS1 for result case 1. 1.4D
	LineSupportForce(support, wantMax)	Returns the maximum or minimum force for the specified line support	LineSupportForce("LS1", true)	Returns the maximum force for line support LS1 across all result cases
	LineSupportMoment(result, support)	Returns the moment for the specified line support for the specified result case	LineSupportMoment("1. 1.4D", "LS1")	Returns the moment for line support LS1 for result case 1. 1.4D
	LineSupportMoment(support, wantMax)	Returns the maximum or minimum moment for the specified line support	LineSupportMoment("LS1", true)	Returns the maximum moment for line support LS1 across all result cases
Report (BACK TO TOP)	AddTable(title)	Adds a specified table to the report	AddTable("Plate Forces")	Adds the Plate Forces table to the report
	SetTableWidth(title, fraction)	Sets the designated table's page width to the specified fraction	SetTableWidth("Vertices", 0.5)	Sets the Vertices table's page width to half
	AddGraphicToReport(title)	Adds the current graphic window with a specified title to the report	AddGraphicToReport("Plate Analysis")	Adds the current graphic window with a title of "Plate Analysis" 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. Simple Model
   b. Refine Mesh

5.4 Example: Simple Model

Refine Mesh

```
//Define and support a simple model.  Then analyze and extract the max displacement.
//The model is a rectangular plate with pinned support on the right and left sides.
//The model is loaded with a uniform pressure.

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

```
ClearLoads();
DeleteAll();

//define the model parameters
var height = 12;
var width = 24;
var thickness = 0.25;
var material = "ASTM A36";
var pressure = 100/12.0/12.0/1000; //100 psf

//build the model
var p1 = new Location(width/2, height/2);
var p2 = new Location(-width/2, height/2);
var p3 = new Location(-width/2, -height/2);
var p4 = new Location(width/2, -height/2);

var boundary = AddBoundary(p1, p2, p3, p4);
Thickness(boundary, thickness);
SetMaterial(material);

//support the model
var s1 = AddLineSupport(p4, p1);
PinLineSupport(s1);
var s2 = AddLineSupport(p3, p2);
PinLineSupport(s2);

//load the model
LoadBoundary(boundary, -pressure, 0, 0);

//mesh refinement would need to be checked, not included here for simplicity
//see the RefineMesh script for an example of that process.
SetMeshFine();
await Analysis();
var disp = Displacement(false);

//Print out the displacement
$"Delta = {disp:f4}"
```

5.5 Example: Refine Mesh

Refine Mesh

```
//Use a loop to refine the mesh of the Simply Supported Plate example project

SetUnits("Kips & Inches");

//open the Simply Supported Plate example before running the script

//Try finer and finer mesh until the extreme MY moment stops changing
var meshSetting = 100;
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
var tolerance = 0.001;  //stop when the moment changes by less than 0.001 k-in/in
while(delta > tolerance)
{
    i++;
    meshSetting *= 2;
    SetMeshCustom(meshSetting);
    await Analysis();
        //interested in the negative moment at the center
    var newMoment = MomentY(false);
    delta = Math.Abs(moment - newMoment);
    moment = newMoment;
}

//Print out the final results
$"MY = {moment:f4}; Final Delta = {delta:f4}; Trials = {i}"
```