PRKNG_sqr / Users Manual

For App Version: 1.0.0.2 / 20220823

App Developers: Woojae Sung @ SA:PN:DA (www.sapnda.com/da)

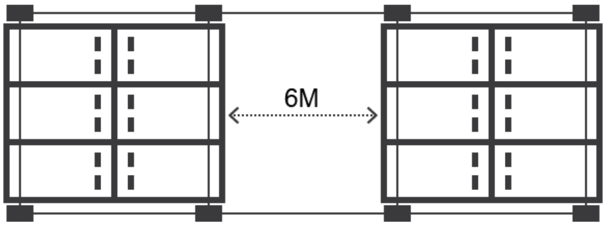
PRKNG_sqr [Parking Square]

Welcome, and thank you for choosing **PRKNG** plugin for Grasshopper!

PRKNG_sqr [Parking Square] calculates the parking stall layout on any given boundary in real-time utilizing a square grid system.

The beauty of the square grid in laying out parking stalls is in its flexibility, bi-directionality, and homogeneity in design, structure, and modular construction perspectives.

In many countries, the square grid system has been widely utilized in underground parking lot planning. Still, the inspiration came from the Korean multi-building apartment complexes with a shared underground parking lot. For example, an 8.1 meters square grid can hold up to six standard(2.5 by 5 meters) car stalls between 600mm wide columns at each corner, leaving at least 6 meters wide driveway between parking grids, which complies with the Korean parking lot regulation.



Parking stall calculation is easy and intuitive: users can set the parking lot boundary and feed some optional inputs, then the plugin calculates the layout in real-time.

Users can also change dimensions (square grid, parking stalls, and columns) per their design intents.

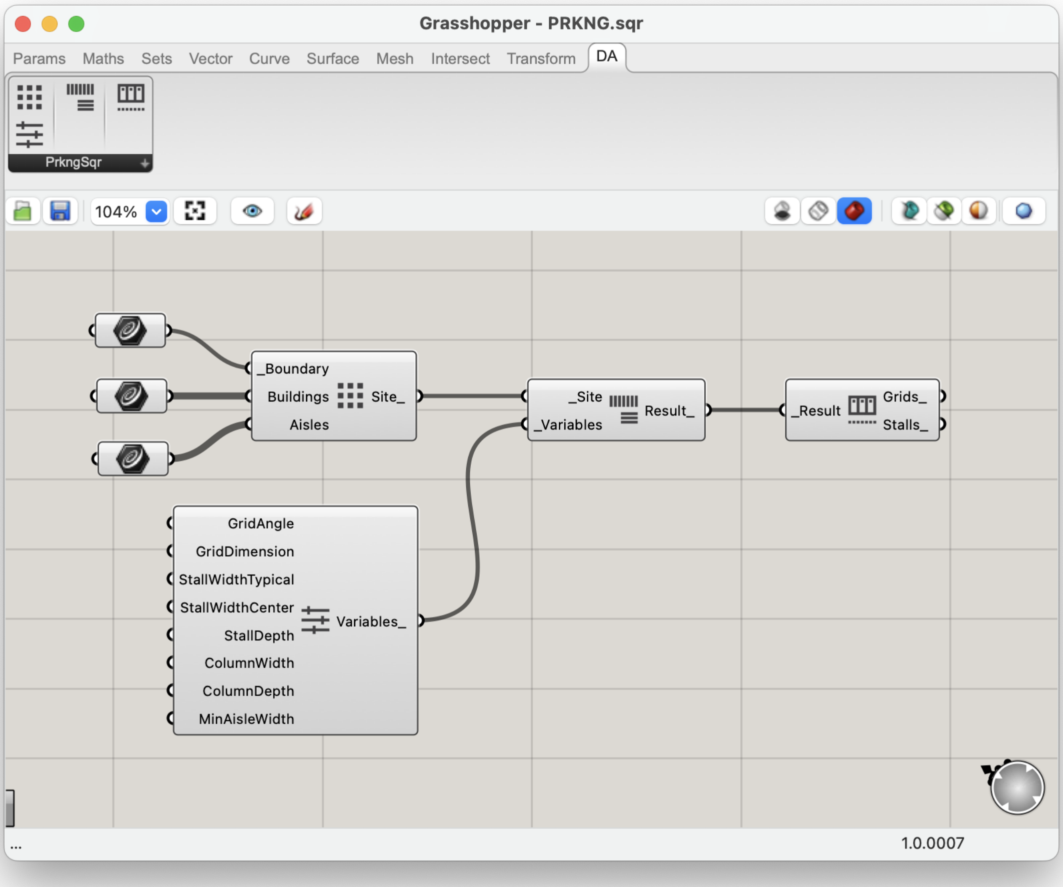
Installation

PRKNG_sqr support both Mac and PC versions of Rhino3d (v7 and above).

There are three ways to install the plugin: through the package manager built in Rhino3d, Food4Rhino/Grasshopper Apps, or manual installation as we used to.

- **Rhino3d built-in package manager**
  - Open Rhino3d, and type in “PackageManager” in the command line.
  - In the search box, look for parking and choose **PRKNG_sqr** from the search result.
  - Then hit the install button, and follow the on-screen instructions.
  - Close and reopen Rhino3d and Grasshopper.
- **Food4Rhino**
  - Go to https://www.food4rhino.com.
  - Go to Grasshopper Apps.
  - Search for parking square.
  - Click download and follow the on screen instruction.
  - For Mac users, this sometimes doesn’t work. Please stick with other installation methods in that case.
- **Manual installation**
  - Although we encourage doing it through the package manager, you still can stick with the old-fashioned.
  - Open Rhino3d and Grasshopper.
  - In the Grasshopper menu, go to File > Special Folders > Components Folder
  - Then copy and paste the downloaded gha file to the folder.
  - Make sure the file is “unblocked” in the file property.
  - Close and reopen Rhino3d and Grasshopper.

Overview



PRKNG_sqr lives under the **DA**[Design-Autonomy] shelf if successfully installed. The standard version has four components: Get Site, Get Variables, Compute Parking, and Visualize Parking. With just four simple steps, one can get the parking layout (two lists of the grid and the stalls) in less than a second.

Components

The Convention

Component variable naming convention, inspired by LB/HB, has been adopted for intuitive usage. Any variable name starting with an underscore is an essential input for the component. One without an underscore means the variable is optional or has a pre-defined default value. Output variable names always end with an underscore.

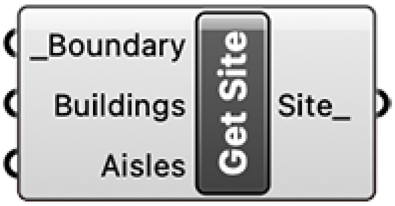
Connecting components is easy enough: match the same variable names (excluding the underscores).

The Unit

The default unit is set to meters. If you are working with other units, you can change variables accordingly at the “Get Variables” component below.

Get Site Information

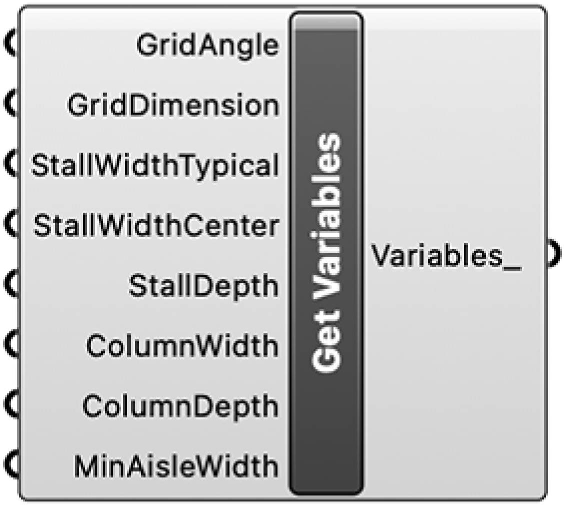
The component collects and sorts the site-related information for the parking calculation.



- **_Boundary**: A closed polyline representing a parking lot boundary. The boundary doesn’t necessarily mean the final parking boundary because the grids may shift slightly while fitting into the given polyline.
- **Buildings**: Optional. A list of closed polylines representing building footprints. The footprints demarcate the buildings from the parking lot.
- **Aisles**: Optional. A list of open/closed polylines representing primary traffic paths.
- **Site_**: The result of the component.

Get Variables

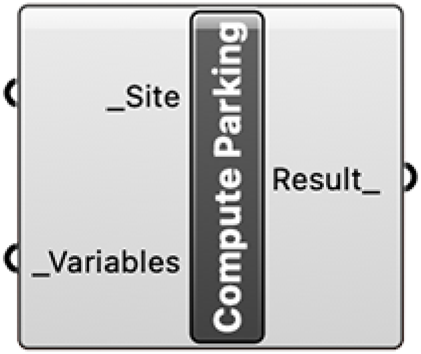
The component defines dimensional variables of the grid, stall, column, and aisle. All the variables of the components are optional, meaning they are all set to the most efficient default values unless you want to change something.



- **GridAngle**: Optional. When working with a non-orthogonal site shape, you might want to align your grid to a specific angle. The default value is 0 degrees.
- **GridDimnsion**: Optional. Depending on the column and stall dimensions, you might need to increase or decrease the grid size. As the plugin utilizes a square grid, you only have one value for the grid.
- **StallWidthTypical** / **StallWidthCenter**: Optional. There are two stall width variables. The stall at the center has a tighter space to open the doors because it doesn't have extra space by the columns. The default value is 2.5 and 2.6 meters, respectively.
- **StallDepth**: Optional. The default value is 5.0 meters.
- **ColumnWidth**: Optional. The default value is 0.6 meters.
- **ColumnDepth**: Optional. The default value is 0.6 meters.
- **MinAisleWidth**: Optional. The Korean building code regulates it to be equal to or wider than 6.0 meters. The default value is 6.0 meters.
- **Variables_**: The result of the component.

Compute Parking Stalls

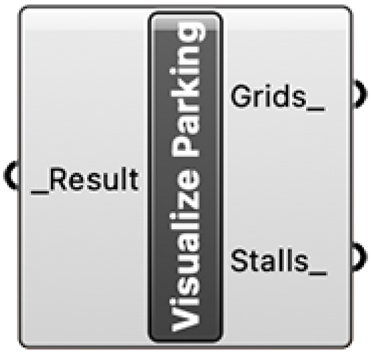
The component computes the parking pattern over the grid.



- **_Site**: Gets the result of the “Get Site” component.
- **_Variables**: Gets the result of the “Get Variables” component.
- **Result_**: The result of the component.

Visualize Parking Stalls

The component interprets the “Compute Parking” calculation result and produces 2D geometries, including grid and stalls.



- **_Result**: Gets the result of the “Compute Parking” component.
- **Grids_**: Returns a list of effective grids.
- **Stalls_**: Returns a list of stalls over the grid.

Feedback / Bug Report

We will be happy to get your feedback! Please let us know if you have any suggestions or any bugs to report. When reporting a bug, please include all the required files to reproduce the issue and send it over to ws@sapnda.com.