PARAMETRIC ENVIRONMENT

THE HANDBOOK OF GRASSHOPPER®
NODES & EXERCISES

Pedro Molina-Siles

EDITORIAL UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ACKNOWLEDGMENTS Manuel Martínez Torán, Pablo Álvarez López, Francisco Javier Navarro Navarro, Lola Navarro Marino de Baldwich, Juan Carlos Piquer Cases, Luis Moreno Serrano, Dennis Sheldon, Débora Cruz, Sarah Levison, Mark Maple, Ihad Matura, Gordon Stone-Levit, Asli Arpak, Selena Karofsky, Shan He, William Román-Saavedra, Martina Papadopoulou, Yu Gao, Philip Aldo Wortmann, Ricardo García Fernández, Javier Bono Cremades, David Bermúdez Martí, Matthew Gazzetes. Department of Expresión Gráfica Arquitectónica of the Universitat Politècnica de València

*The first part of this publication [*Nodes + Complements + Remarks*] has been possible thanks to the project of the International Plan of High Technologies entitled *Study and parametric analysis for the development of a knowledge network generative studies through the 4K-3.0 technology* (USA-M, 2014-201006-X), funded by Computation Group and Digital Design & Fabrication Group, research teams belonging to the Department of Architecture of the MIT

To reference this publication use the following quote:
MOLINA-SILES, P. (2016). *Parametric Environment. The handbook of Grasshopper®. Nodes & Exercises.* Universitat Politècnica de València: Valencia

The contents of this publication has been reviewed by double blind system, following the procedure set out in:
http://www.upv.es/entidades/AEUPV/info/891747normalc.html

AUTHOR Pedro Molina-Siles

TEXT EDITING [*Nodes*]
[Universitat Politècnica de València]: Pedro Molina-Siles
[Massachusetts Institute of Technology]: Dennis Sheldon, Débora Cruz, Luis Moreno Serrano, Sarah Levison, Mark Maple, Ihad Matura, Gordon Stone-Levit, Asli Arpak, Selena Karofsky, Shan He, William Román-Saavedra, Martina Papadopoulou, Yu Gao, Philip Aldo Wortmann

DESIGN [*Exercises*]
[Universitat Politècnica de València]: Pedro Molina-Siles, Ricardo García Fernández, Javier Bono Cremades, David Bermúdez Martí
[Massachusetts Institute of Technology]: Matthew Gazzetes

ORIGINAL DESIGN AND LAYOUT Pedro Molina-Siles, Juan Carlos Piquer Cases
PICTURE ON COVER AND INSIDE PAGES *Nodes*. Watercolor by Hugo Barros Costa
PRINTING Byprint Percom, sl
PUBLISHER Universitat Politècnica de València

Distribution: Phone. (+34) 963 877 012 / www.lalibreria. upv.es / Ref.: 0270_03_01_01

ISBN: 978-84-9048-499-9

Printed on demand

Contact: pmolina@ega.upv.es / pmolinasiles@gmail.com

FOREWORD

FORGET ABOUT SPECIFIC RESULTS; JUST CONSIDER THE PROCESS

Parametric design enables the generation of geometric shapes by defining an initial set of parameters. In turn, the parameters establish formal relationships between each other. In other words, it is a question of creating a hierarchy of mathematical and geometric relationships that generate a particular design. In this parametric environment, the design process enables us to explore the entire range of possibilities that the initial parameters afford us. What does this mean? When you are asked by a client to come up with different versions of a test design, the required effort is minimized on a massive scale. Why? Because what we have here is an automated procedure that does away with tedious repetitive tasks and the need for complicated calculations on the fly and it also eliminates human error. Even slight changes to the starting parameters can lead to significant changes in output. With parametric design, focus is somewhat taken away from the appearance, and focuses instead on the conduct: forget about how it appears, but rather how it performs; forget about static, defined solutions, and think about the design of phases and specific elements; forget about a single solution, and think about a collection of possible outcomes. Presently, version 0.9.0076 of the Grasshopper® software enables you to move from a solitary solution to an indefinite number of them.

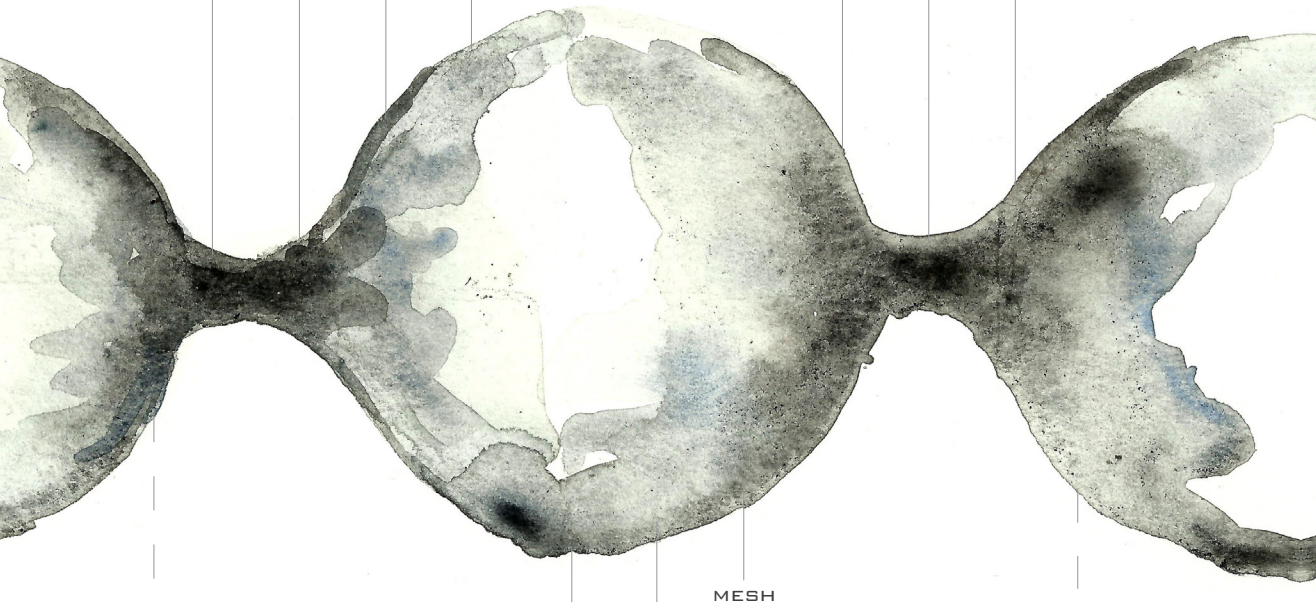
Grasshopper® is a visual programming editor developed by David Rutten at Robert McNeel & Associates. As a plug-in for Rhinoceros 3D®, Grasshopper® is integrated with the robust and versatile modeling environment used by creative professionals across a diverse range of fields, including architecture, engineering, product design, and more. In tandem, Grasshopper® and Rhino offer us the opportunity to define precise parametric control over models, the capability to explore generative design workflows, and a platform to develop higher-level programming logic – all within an intuitive, graphical interface. The origins of Grasshopper® can be traced back to the functionality of Rhinoceros 3D® Version 4's "Record History" button. This built-in feature enabled users to store modeling procedures implicitly in the background along the way. If you lofted four curves with the recording on and then edited the control points of one of these curves, the surface geometry would update. Back in 2008, David posed the question: "what if you could have more explicit control over this history?" and the precursor to Grasshopper®, Explicit History, was born. This exposed the history tree to editing in detail and empowered the user to develop logical sequences beyond the existing capabilities of Rhino3D's built in features. Six years later, Grasshopper® is now a robust visual programming editor that can be extended by suites of externally developed add-ons. Furthermore, it has fundamentally altered the workflows of professionals across multiple industries and fostered an active global community of users.

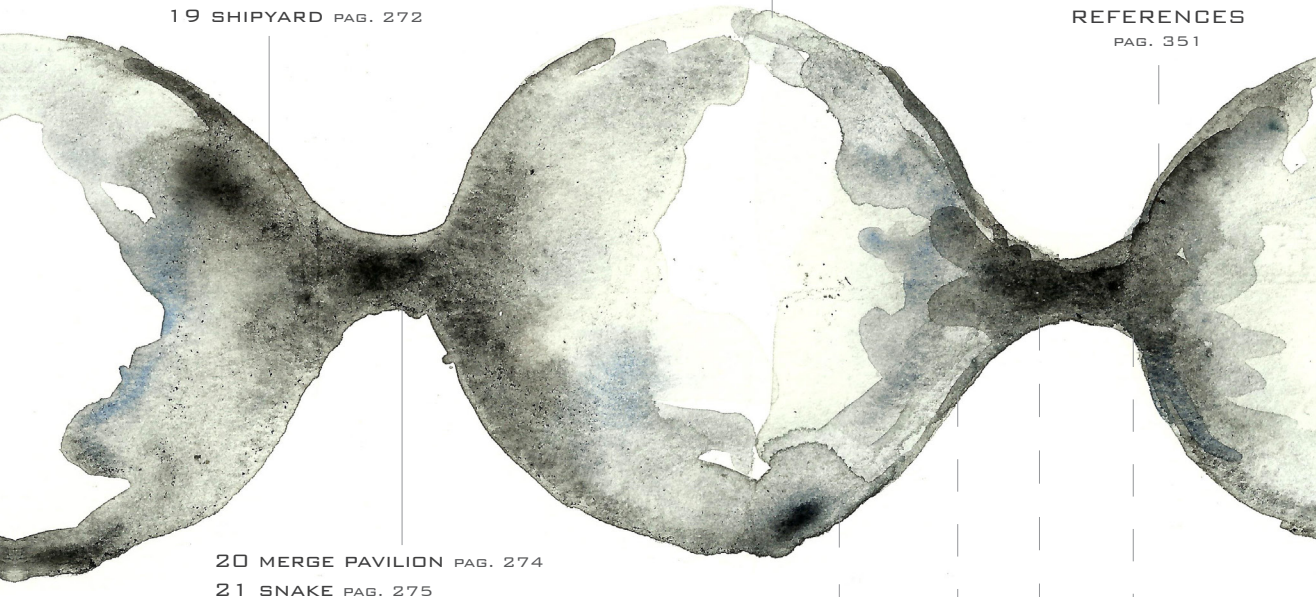
Parametric Environment. The handbook of Grasshopper®. *Nodes & Exercises* aims, in a clear and concise manner, to simplify access to Grasshopper® in two ways. Firstly, to classify each of its building-block components: Params, Maths, Sets, Vector, Curve, Surface, Mesh, Intersect, Transform and Display. Our attention is focused on examining each of these components; its purpose as well as all the parameters provided by the input and output data that are so essential to this plug-in. Secondly, to apply some of these nodes to a series of exercises.

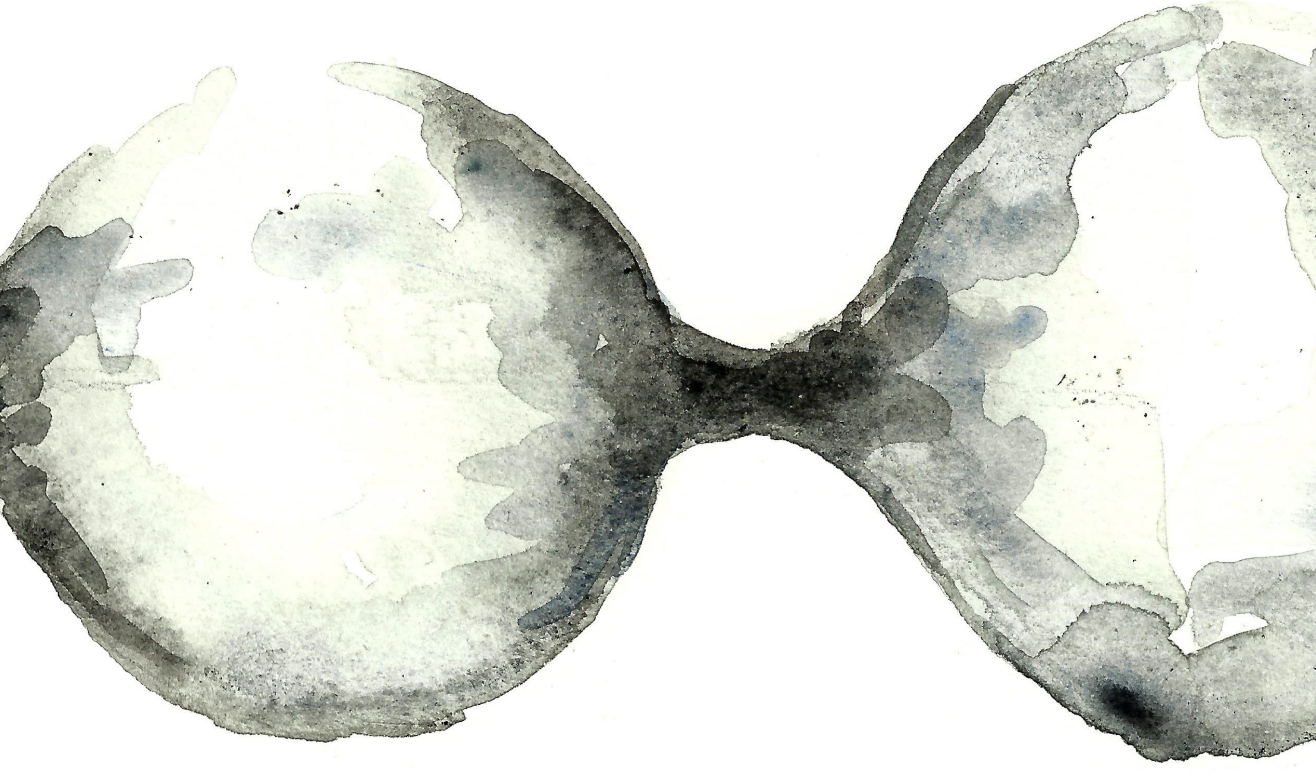
With Grasshopper®, forget about specific results; just consider the process.

Pedro Molina-Siles / Dennis Sheldon

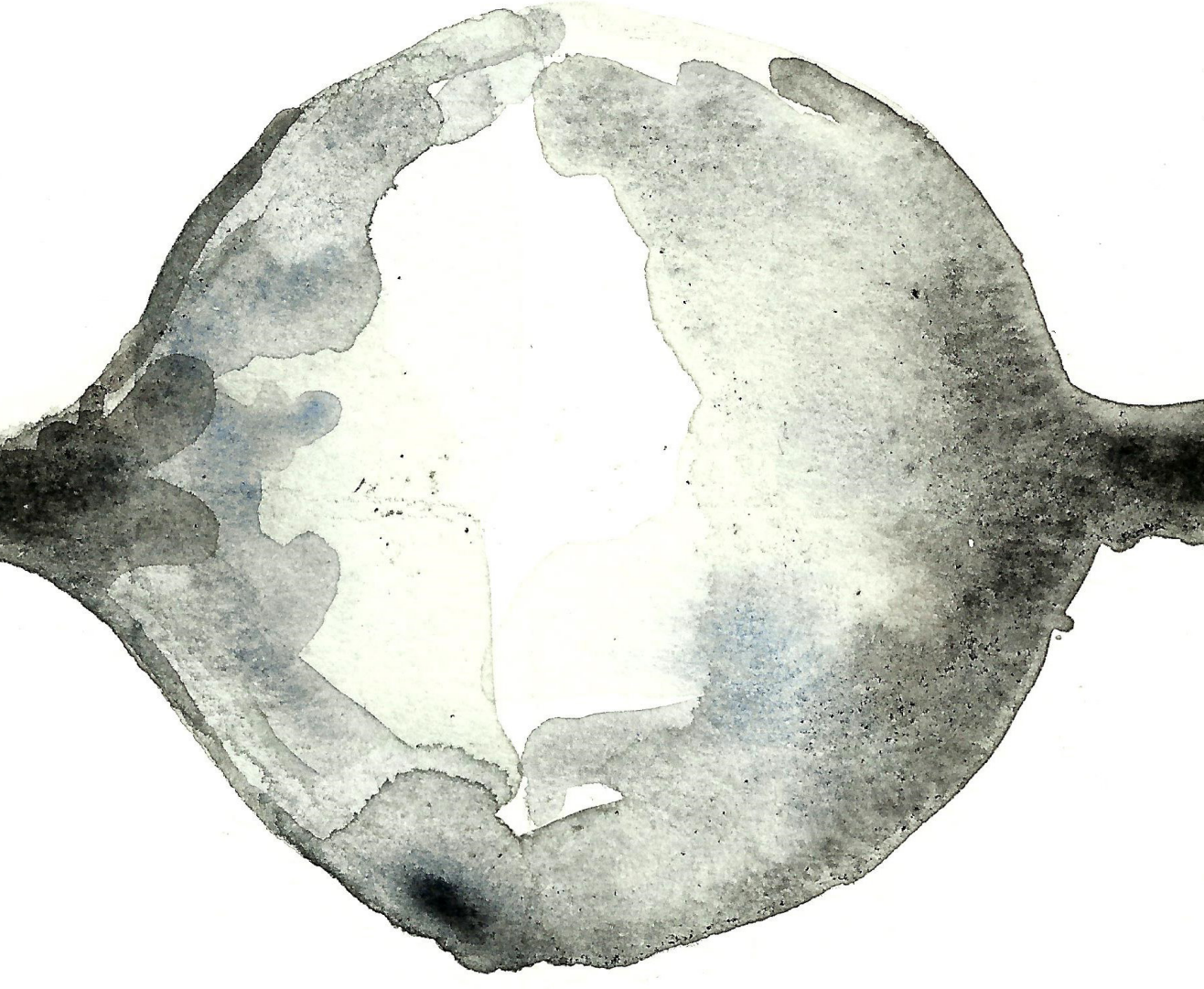
CONTENTS

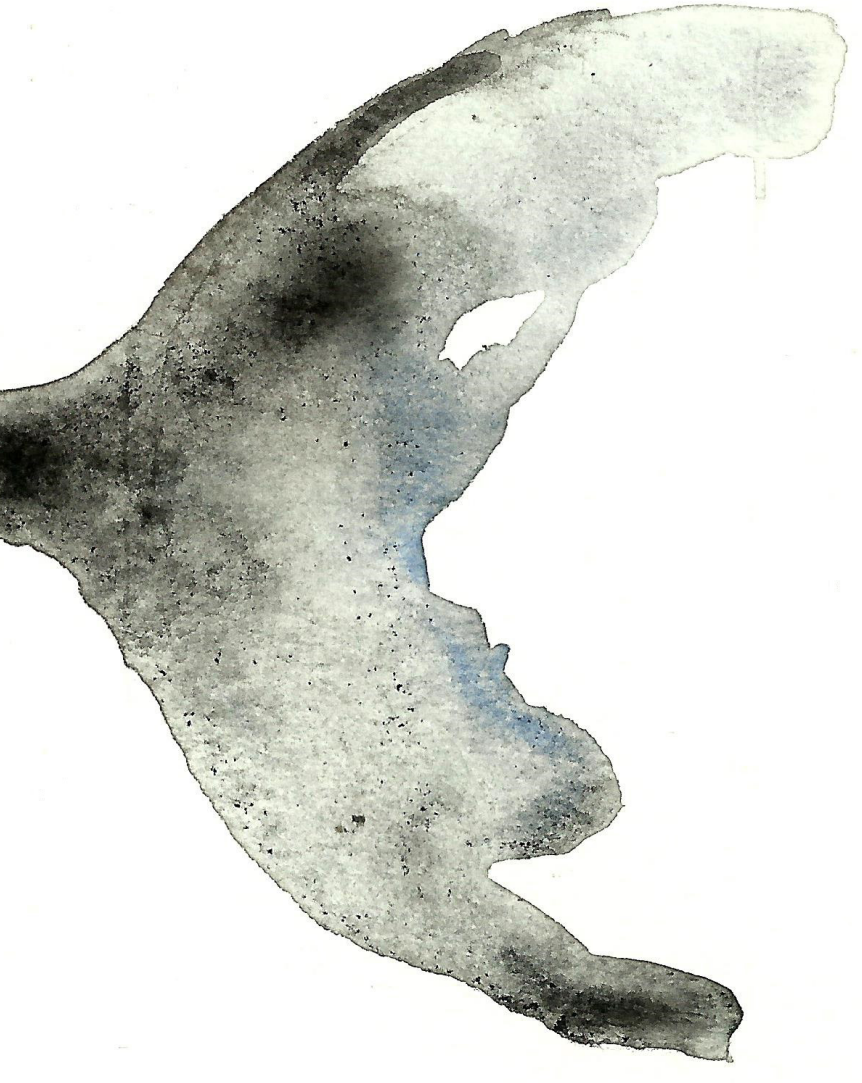






NODES





PARAMS

POINT (Pt)

Represents a collection of 3D Point coordinates.

Point parameters are capable of storing persistent data. You can set the persistent records through the parameter menu. Remarks p. 317

Exercises: 02,17,18,20,21,22,23,24,25,26,27,28,31,35,37,40,42, 43,48,49.

VECTOR (Vec)

Represents a collection of 3D Vectors. Vectors are interchangeable with Points, but for purposes of clarity they have their own parameter type.

Vector parameters are capable of storing persistent data. You can set the persistent records through the parameter menu. Remarks p. 317

Exercises: 12.

CIRCLE

Represents a collection of Circle primitives. Circles are stored as parametric (non-nurbs) curves internally and can thus not be linked to circular curve objects in the Rhino model since circles in Rhino are not guaranteed to remain circles; both linear transforms and morphs might turn them into ellipses or nurbs curves.

Circle parameters are capable of storing persistent data. You can set the persistent records through the parameter menu. Remarks p. 318

CURVE (Crv)

Represents a collection of Curve geometry. Curve geometry is the common denominator of all curve types in Grasshopper. Remarks p. 318

PLANE (Pln)

Represents a collection of Plane primitives. Planes are defined by an origin point and three axis vectors. They represent local coordinate-systems which are often used in transformation operations.

Plane parameters are capable of storing persistent data. You can set the persistent records through the parameter menu. Remarks p. 318
Exercises: 34.

CIRCULAR ARC (Arc)

Represents a collection of Circular Arc primitives. Arcs are stored as parametric (non-nurbs) curves internally and can thus not be linked to arc-like curve objects in the Rhino model since arcs in Rhino are not guaranteed to remain arcs; both linear transforms and morphs might turn them into elliptical arcs or nurbs curves.

Arc parameters are capable of storing persistent data. You can set the persistent records through the parameter menu. Remarks p. 319

LINE

Represents a collection of Line primitives. Lines are stored as parametric (non-nurbs) curves internally and can thus not be linked to linear curve objects in the Rhino model since lines in Rhino are not guaranteed to remain lines.

Line parameters are capable of storing persistent data. You can set the persistent records through the parameter menu. Remarks p. 319
Exercises: 35, 38,47.

RECTANGLE (Rec)

Represents a collection of Rectangle primitives. Rectangles are stored as parametric (non-nurbs) curves internally and can thus not be linked to rectangular curve objects in the Rhino model since rectangles in Rhino are not guaranteed to remain rectangles; both linear transforms and morphs might turn them into parallelograms or nurbs curves.

Rectangle parameters are capable of storing persistent data. You can set the persistent records through the parameter menu. Remarks p. 319

BOX

Represents a collection of oriented Box geometry.

Box parameters are capable of storing persistent data. You can set the persistent records through the parameter menu. Remarks p. 319
Exercises: 04.

MESH

Represents a collection of Mesh geometry. Meshes in Rhino consist solely of triangles and quads. Meshes always contain a vertex array, a normal array (one for each vertex) and a face array. Meshes can optionally contain vertex texture coordinates and colours as well. Remarks p. 320

SURFACE (Srf)

Represents a collection of Surface geometry. Surface geometry is the common denominator of all surface types in Grasshopper. Remarks p. 320
Exercises: 25,41.

BREP

Represents a collection of Brep geometry. Brep stands for 'Boundary REPresentation' and all surfaces and polysurfaces in Rhino are Breps. If a Brep has only one face, it is considered a surface in Grasshopper. Remarks p. 320

MESH FACE (Face)

Represents a collection of 3D mesh faces. Mesh face parameters are a utility type that is used for mesh components. It exposes almost no user control. Remarks p. 321

TWISTED BOX (TBox)

Represents a list of twisted boxes. Twisted boxes are primarily used in deformation (morph) components where they define non-euclidean spaces.

Twisted Box parameters are capable of storing persistent data. You can set the persistent records through the parameter menu. Remarks p. 321

FIELD

Contains a collection of vector fields. Remarks p. 321

GEOMETRY CACHE

Bake or Load geometry to and from the Rhino document. Remarks p. 321

GROUP (Grp)

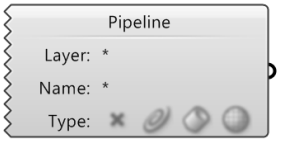
Represents a collection of grouped geometry.
Remarks p. 321

GEOMETRY (Geo)

Represents a collection of 3D Geometry. Since new types of geometry are continually added, the preview and conversion algorithms of this parameter may be incomplete as it fails to recognize some data types that could -in a perfect world- have been treated like regular geometry.
Remarks p. 322
Exercises: 04,10,19,20,32,36,37,44,48,49.

GEOMETRY PIPELINE (Pipeline)

Defines a geometry pipeline from Rhino to Grasshopper. Remarks p. 322



TRANSFORM

Represents a collection of three-dimensional linear transforms. Transforms can consist of both affine transformations such as translation and rotations and non-affine transformations such as shearing and tapering. Remarks p. 322

BOOLEAN (Bool)

Represents a collection of Boolean (True/False) values. Boolean parameters are capable of storing persistent data. You can set the persistent records through the parameter menu. Remarks p. 322

NUMBER (Num)

Represents a collection of double-precision floating point values (it has got nothing to do with two of anything). Double parameters are capable of storing persistent data. You can set the persistent records through the parameter menu. Remarks p. 322
Exercises: 17,35,44.

INTEGER (Int)

Represents a collection of Integer numeric values. Integer parameters are capable of storing persistent data. You can set the persistent records through the parameter menu. Remarks p. 323
Exercises: 44.

TEXT (Text)

Represents a collection of Text fragments. String parameters are capable of storing persistent data. You can set the persistent records through the parameter menu. Remarks p. 323
Exercises: 34.

COLOUR (Col)

Represents a collection of colour values. Colour parameters are capable of storing persistent data. You can set the persistent records through the parameter menu. Remarks p. 323

CULTURE

Contains a collection of culture specifiers.
Remarks p. 323

DOMAIN²

Contains a collection of two-dimensional domains. 2D Domains are typically used to represent surface fragments. A two-dimensional domain consists of two one-dimensional domains. Remarks p. 323

MATRIX

Contains a collection of numeric matrices.
Remarks p. 324

COMPLEX (C)

Represents a collection of complex numbers. Complex parameters are capable of storing persistent data. You can set the persistent records through the parameter menu. Remarks p. 324

DOMAIN

Represents a collection of one-dimensional Domains. Domains are typically used to represent curve fragments and continuous numeric ranges.

A domain consists of two numbers that indicate the limits of the domain, everything in between these number is part of the domain. Remarks p. 324

GUID (ID)

Represents a collection of Guids. Guid parameters are capable of storing persistent data. You can set the persistent records through the parameter menu. Remarks p. 324

TIME

Represents a collection of Time and Date values. Time parameters are capable of storing persistent data. You can set the persistent records through the parameter menu. Remarks p. 325

DATA

Represents a collection of... well, anything really. This parameter type will happily eat whatever you decide to feed it. As a result, the preview of this parameter may be incomplete as it might fail to recognize some data types that could -under ideal circumstances- have been displayed in the viewport. Remarks p. 325

FILE PATH (Path)

Contains a collection of file paths. Remarks p. 325

DATA PATH (Path)

Represents a collection of Data Tree branch paths. Grasshopper stores data in hierarchical lists not dissimilar to a branching tree structure. Every branch in the data tree is defined by a series of index integers. Path parameters are capable of storing persistent data. You can set the persistent records through the parameter menu. Remarks p. 325

SHADER

Represents a collection of Shader (shading material) values. Shaders are used during real-time and render-time display of geometry. The Grasshopper shader is based on the Rhino basic material which supports a number of settings most of which can be viewed real-time in the viewport.

In addition, Grasshopper shaders can also reference RDK materials, which can contain procedural and bitmap textures and nested sub-shaders. Note however that all advanced RDK shaders are downgraded to Rhino basic materials during OpenGL drawing. If you do not have the RDK installed (or if you have an old version), the RDK menu option will be disabled. Remarks p. 325

NUMBER SLIDER ()

A slider is a special interface object that allows for quick setting of individual numeric values. You can change the values and properties through the menu, or by double-clicking a slider object. Sliders can be made longer or shorter by dragging the rightmost edge left or right. Note that sliders only have output grips. Sliders appear automatically in the Grasshopper Panel. Remarks p. 325
Put into practice on all exercises.

PANEL ()

A Panel is like a Post-It™ sticker. It is typically an inactive object that allows you to add little remarks or explanations to a Document. You can change the text through the menu or by double-clicking the panel surface.

You can change the font that a Panel uses through the menu. There are some predefined fonts, but you can also specify a custom font. Be aware that if you pick a non-standard Font, the panel might not display correctly when loaded on a system which lacks that font.

Panels can also receive their information from elsewhere. If you plug an output parameter into a Panel, you can see the contents of that parameter in real-time. All data in Grasshopper can be viewed in this way.

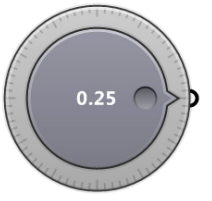
Panels can also stream their content to a textfile. You can enable/disable streaming through the Panel menu. Be aware that the textfile will be overwritten whenever the Panel content updates. You should not open these textfiles in applications which set readonly attributes while the source Panel is still active. Content streaming always creates a simple textfile, the different extensions that are available in the Stream Path window do not affect in any way the content of the file. If you need formatting, you should consider running your data through an Expression or Script component first. Remarks p. 326
Exercises: 01,02,04,08,17,19,22,25,27,28,30,34,37,38,39,44,46, 47,49.

BOOLEAN TOGGLE (Toogle)

Boolean (true/false) toggle. Remarks p. 326
Exercises: 00,07,13,20,22,27,37,38,39,49.

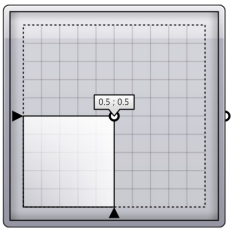
CONTROL KNOB (Knob)

A radial dial knob for settings numbers.
Remarks p. 326



MD SLIDER

A multidimensional slider. Remarks p. 326
Exercises: 25.



BUTTON

Button object with two values. Remarks p. 326

DIGIT SCROLLER ()

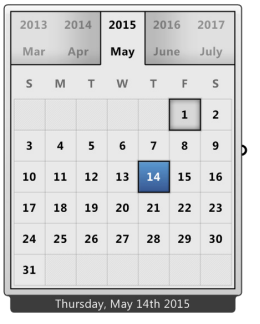
A scroller is a special interface object that allows for quick setting of individual numeric values. You can change the values by dragging the digits up or down, by dragging the radix point left and right, by clicking on the sign symbol or by double clicking and entering a numeric expression. Note that scroller only have output grips.

VALUE LIST (List)

Provides a list of preset values to choose from.
Remarks p. 326

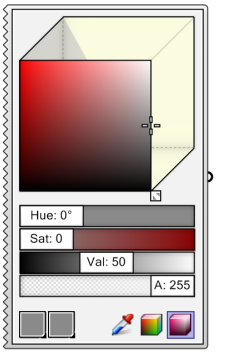
CALENDAR ()

Represents a calendar.



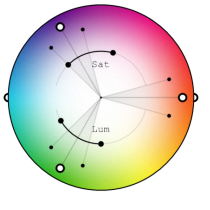
COLOUR PICKER (Colour)

Provides a colour picker object. Remarks p. 326
Exercises: 34.

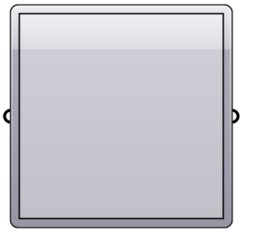


COLOUR WHEEL

Creates a palette of related colours. Remarks p. 326



GRAPH MAPPER (Graph)



Graph mapper objects allow you to remap a set of numbers. By default the {x} and {y} domains of a graph function are unit domains (0.0 ~ 1.0), but these can be adjusted via the Graph Editor. Graph mappers can contain a single mapping function which can be picked through the context menu. Graphs typically have grips (little circles) which can be used to modify the variables that define the graph equation.

By default, a graph mapper objects contains no graph and performs a 1:1 mapping of values. Remarks p. 326
Exercises: 18,48.

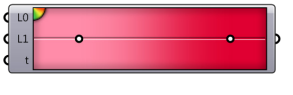
CLOCK ()

Represents a 24 hour clock.

COLOUR SWATCH

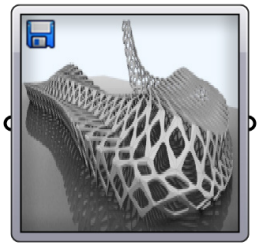
A swatch is a special interface object that allows for quick setting of individual colour values. You can change the colour of a swatch through the context menu. Remarks p. 327
Exercises: 32,34,35,40.

GRADIENT



Gradient controls allow you to define a colour gradient within a numeric domain. By default the unit domain (0.0 ~ 1.0) is used, but this can be adjusted via the L0 and L1 input parameters. Remarks p. 327

IMAGE SAMPLER (Image)



The image sampler object allows you to evaluate pixel data stored in image files. You can instantiate an Image Sampler simply by dragging an image file from Windows Explorer onto a Grasshopper canvas. If you want to change the referenced file path, you can do so via the menu or the Image Sampler Settings dialog. The Image Sampler object has several options that pertain to the sampling algorithm, they can all be set via the context menu or the aforementioned Settings dialog:

Interpolate: If Interpolate is enabled, then images are sampled using floating point pixel coordinates. Values in between pixels are blended linearly.
Clamp: This option results in completely transparent colours when the image is sampled beyond the bounds.
Tile: This option will repeat the image ad infinitum in all directions.
Flip: This option will repeat and mirror the image ad infinitum in all directions.
X and Y domains: Defines the sampling range in both directions. By default the sampling range is {0.0 to 1.0} in both x and y.

The preview image on the Image Sampler object will be degraded if the source file contains more than 40,000 pixels. This sampling algorithm always operates on the original data. Remarks p. 327

ATOM DATA (Atom)

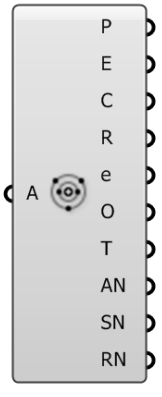
Get detailed information for an atom.

INPUT parameters:

A	(Atom)	Atom to evaluate.

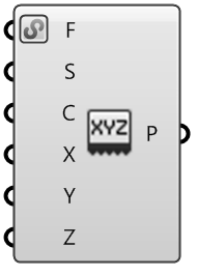
OUTPUT parameters:

P	(Point)	Location of atom.
E	(Text)	Element name of atom.
C	(Text)	Chain ID to which this atom belongs.
R	(Text)	Residue name to which this atom belongs.
e	(Integer)	Charge of this atom.
O	(Number)	Occupancy of this atom.
T	(Atom)	Temperature factor of this atom.
AN	(Integer)	Atomic number of atom.
SN	(Integer)	Atom serial number.
RN	(Integer)	Residue serial number.



IMPORT COORDINATES (Coords)

Import point coordinates from generic text files.



INPUT parameters:

- **F** (Text) Location of point text file.
- **S** (Text) Coordinate fragment separator.
- **C** (Text) Optional comment line start.
- **X** (Integer) Index of point X coordinate.
- **Y** (Integer) Index of point Y coordinate.
- **Z** (Integer) Index of point Z coordinate.

OUTPUT parameters:

- **P** (Point) Imported points.

IMPORT PDB (PDB)

Import data from Protein Data Bank *.pdb files.

INPUT parameters:

- **F** (Text) Location of *.pdb file.

OUTPUT parameters:

- **A** (Atom) All atoms in the PDB file.
- **B** (Line) Bonds between atoms.

READ FILE (File) -Per Line-

Read the contents of a file.

INPUT parameters:

- **F** (Text) Uri or file to read.

OUTPUT parameters:

- **C** (Generic Data) File content.

IMPORT 3DM (3DM)

Import geometry from Rhino 3dm files.

INPUT parameters:

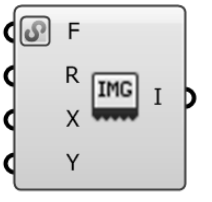
- **F** (Text) Location of Rhino 3dm file.
- **L** (Text) Layer name filter.
- **N** (Text) Object name filter.

OUTPUT parameters:

- **G** (Geometry) Imported geometry.

IMPORT IMAGE (IMG)

Import image data from bmp, jpg or png files.



INPUT parameters:

- **F** (Text) Location of image file.
- **R** (Rectangle) Optional image destination rectangle.
- **X** (Integer) Number of samples along image X direction.
- **Y** (Integer) Number of samples along image Y direction.

OUTPUT parameters:

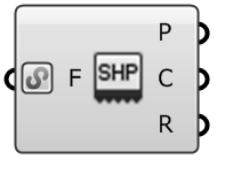
- **I** (Mesh) A mesh representation of the image.

IMPORT SHP (SHP)

Import data from GIS *.shp files.

INPUT parameters:

- **F** (Text) Location of *.shp file.



OUTPUT parameters:

- **P** (Point) Points in file.
- **C** (Curve) Curves in file.
- **R** (Brep) Regions in file.

CHERRY PICKER (Item)

Pick a single item from a data tree. Remarks p. 328

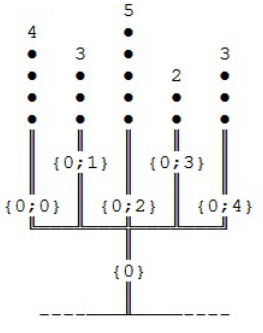
JUMP

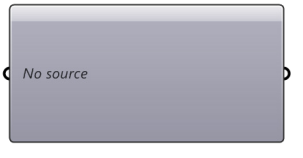
Just between different locations.

PARAM VIEWER (Viewer)

As of Grasshopper 0.6, data can be stored in hierarchical structures not dissimilar to a branching tree. Data is still stored in lists, but each list now has a 'path', which is a series of indices that describe the position of the data branch inside the tree.

For example, if we divide 5 curves by length, each curve might give us a different amount of division points. All 4 points that originate from the first curve will be stored in a branch at path {0;0} and all 5 points that originate from the third curve will be stored at path {0;2} :





Note that Branch {0} contains no data and is thus omitted from the data structure.
The representation of this data tree in the param viewer will be:
Structure (Paths = 5)
path {0;0} (N = 4)
path {0;1} (N = 3)
path {0;2} (N = 5)
path {0;3} (N = 2)
path {0;4} (N = 3)
Remarks p. 328

SCRIBBLE
Represents a quick note.

Doubleclick Me!

DATA DAM
Data is already up to date, you do not need to click here.

INPUT parameters:

A (Data A (A) as tree) Data to buffer.

OUTPUT parameters:

A (Data A (A) as tree) Buffered data.

DATA RECORDER (Rec)
Records data over time. Remarks p. 328

TIMER

Timers are object which fire update events at specified intervals. This process is reasonably dangerous since updates might occur when you do not expect them, so please be careful when using them, and only use a timer when you have no other option.

By default a new timer enabled, but inactive as it is isolated. You can enable/disable a timer by double clicking on the object or via the context menu. However, enabling a timer is no guarantee that it will fire update events.

First, there is a global Timer Abort which has the power to disable all timers in Grasshopper. Whenever a timer is enabled for the first time, the Global Abort will appear in the Windows notification bar. When the icon is green, it means the Global Abort is off and timers are allowed to fire events. When the icon is red, all timers are blocked. Blocked timers are displayed with a red icon instead of the timer icon on the canvas. Double clicking on the notification icon on the windows taskbar will toggle the Global Abort state.

Secondly, timers only fire events when they can make a difference. Before a timer will tell Grasshopper to recompute the solution it will blank certain objects. These are called the targets of the timer object. You can add a target to a timer by click+dragging from the arrow area to the right of the timer. Drag the wire onto another object, and it will be added to the target list. You can remove objects from the target list by tracing over an existing target wire while holding the Control key. Remarks p. 328

CLUSTER INPUT

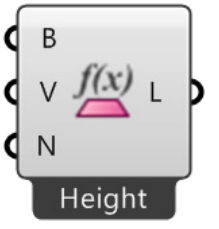
Represents a cluster input parameter.

CLUSTER OUTPUT

Represents a cluster output parameter.

FITNESS LANDSCAPE (LScape) -Height-

Display a 2.5D fitness landscape.



INPUT parameters:

- **B** (Rectangle) Landscape bounds.
- **V** (Number) Landscape values.
- **N** (Integer) Number of samples along X direction.

OUTPUT parameters:

- **L** (Mesh) Landscaper mesh.

GALAPAGOS Evolutionary Solver

Double click to open the Galapagos editor.

INPUT parameters:

- **Genome** Define all the sliders that are part of the Genome.
- **Fitness** Define the numbers that represent the firness.

Exercises: 17.

Para seguir leyendo haga click aquí