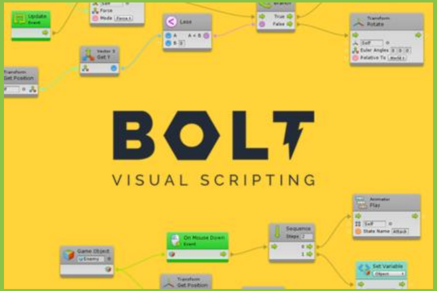
KINE 458: Virtual Interactive Worlds

Logic and Unity Components



Todays Objectives:

- Review
- Introduction to Unity Object Components
- Introduction to Programming Logical Operations
- Bolt Visual Scripting with Logic and Components

Virginia Commonwealth University
Gavin Alberghini
(alberghinig@vcu.edu)

Bolt Review

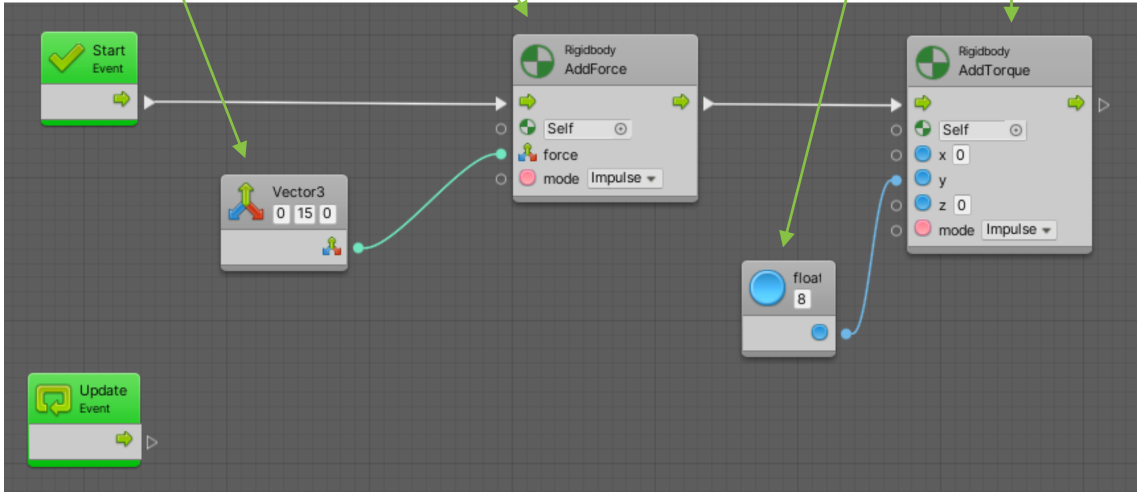
❑ Visual Scripting is a visual way of creating programs for your objects in Unity.

Method:
- Adds a force vector to an object
- Must specify force type and RigidBody

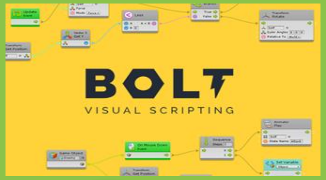
Variable:
- Vector 3
- Object Type

Method:
- Adds a rotational force vector to an object
- Must specify force type and RigidBody

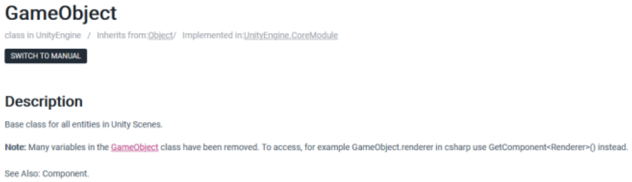
Variable:
- Float literal
- Primitive Type



Reading the Unity Application Programming Interface



Description – What is the general purpose of some object, component, etc?



GameObject

class in UnityEngine / Inherits from:Object/ Implemented in:UnityEngine.CoreModule

SWITCH TO MANUAL

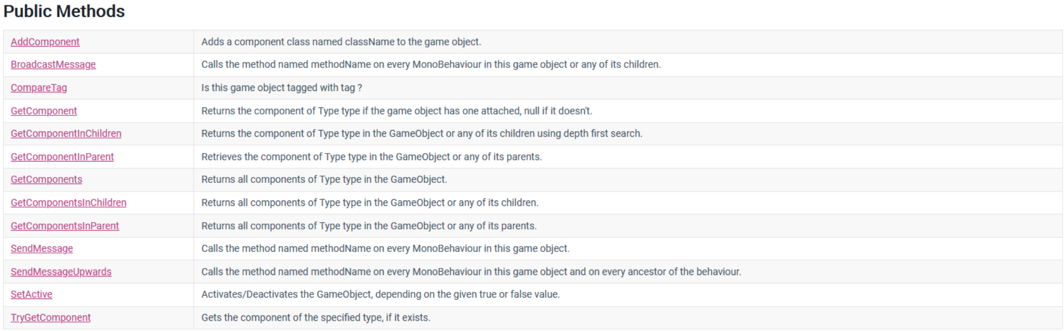
Description

Base class for all entities in Unity Scenes.

Note: Many variables in the GameObject class have been removed. To access, for example GameObject.renderer in csharp use GetComponent<Renderer>() instead.

See Also: Component.

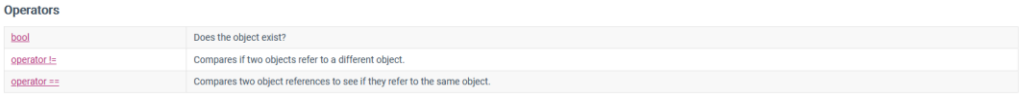
Public Methods – What functionality already exists with this object, component, etc?



Public Methods

AddComponent	Adds a component class named className to the game object.
BroadcastMessage	Calls the method named methodName on every MonoBehaviour in this game object or any of its children.
CompareTag	Is this game object tagged with tag ?
GetComponent	Returns the component of Type type if the game object has one attached, null if it doesn't.
GetComponentInChildren	Returns the component of Type type in the GameObject or any of its children using depth first search.
GetComponentInParent	Retrieves the component of Type type in the GameObject or any of its parents.
GetComponents	Returns all components of Type type in the GameObject.
GetComponentsInChildren	Returns all components of Type type in the GameObject or any of its children.
GetComponentsInParent	Returns all components of Type type in the GameObject or any of its parents.
SendMessage	Calls the method named methodName on every MonoBehaviour in this game object.
SendMessageUpwards	Calls the method named methodName on every MonoBehaviour in this game object and on every ancestor of the behaviour.
SetActive	Activates/Deactivates the GameObject, depending on the given true or false value.
TryGetComponent	Gets the component of the specified type, if it exists.

Operators – What can I utilize to evaluate this object?

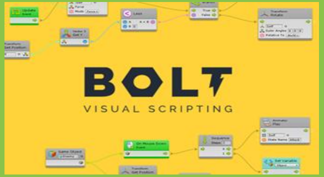


Operators

bool	Does the object exist?
operator !=	Compares if two objects refer to a different object.
operator ==	Compares two object references to see if they refer to the same object.

Logic / Components

Unity



GameObjects – The basic class of any object in a Unity scene.

Vector3 – An object representing a 3-dimensional vector of primitive datatypes. Typically ints or floats.

Transform – A component representing position, orientation, and scale as 3 separate Vector3s.

RigidBody – A component that adds a game object to Unity’s physics engine. Has many built in functions for intractability.

Collider – A component that registers when the owning game object collides with another object.

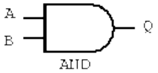
Prepositional Logic

[Introduction to Prepositional Logic](#)

[Truth Tables](#)

These conditional (logical) statements will help guide the execution of your flow graphs by allowing you to have dynamic control over which portions of the graph are executed.

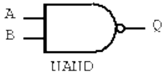
Java Symbol	Operator
==	Equal To (See Caution Below)
!=	Not Equal To
<	Less Than
<=	Less Than Or Equal To
>	Greater Than
>=	Greater Than or Equal To
&&	AND
||	OR



$Q = A \cdot B$

AND

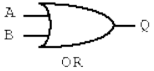
A	B	Q
0	0	0
0	1	0
1	0	0
1	1	1



$Q = \overline{A \cdot B}$

NAND

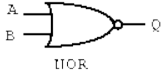
A	B	Q
0	0	1
0	1	1
1	0	1
1	1	0



$Q = A + B$

OR

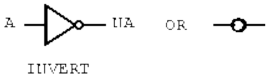
A	B	Q
0	0	0
0	1	1
1	0	1
1	1	1



$Q = \overline{A + B}$

NOR

A	B	Q
0	0	1
0	1	0
1	0	0
1	1	0

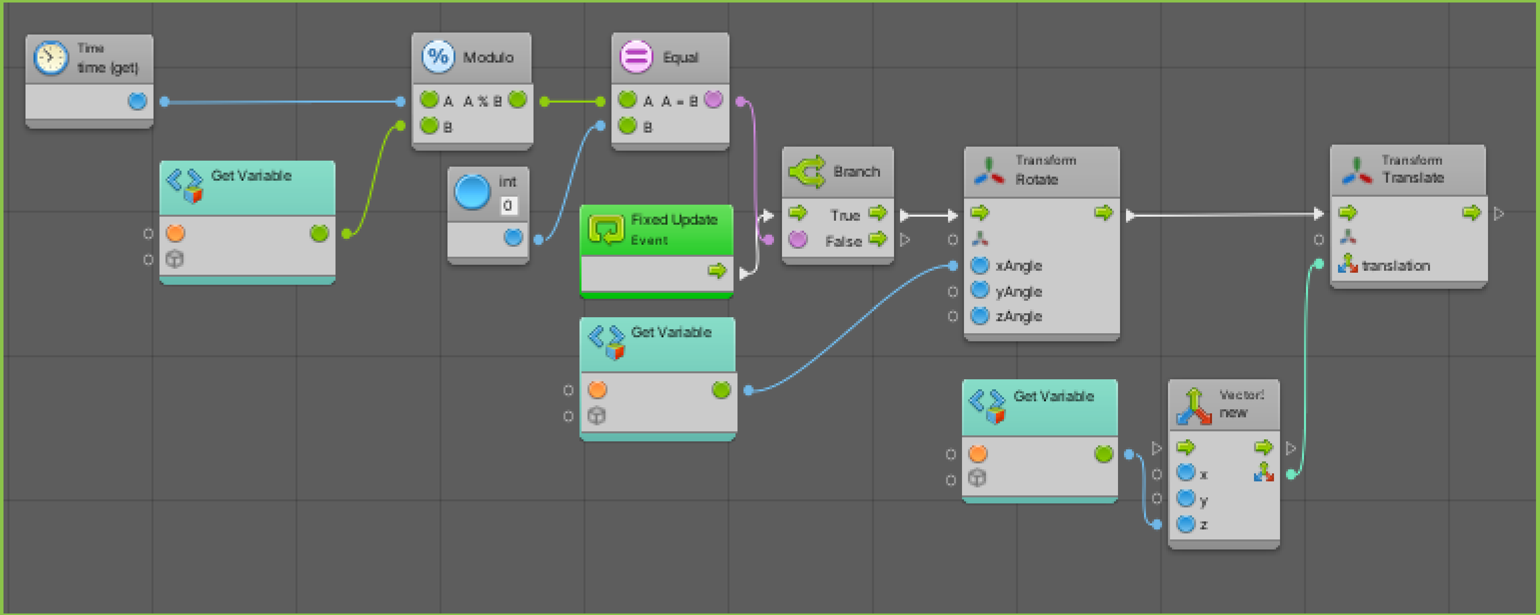


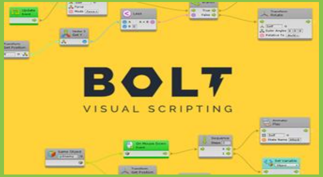
$Q = \overline{A}$

NOT

A	Q
0	1
1	0

Section Two: Live Demo





Questions?