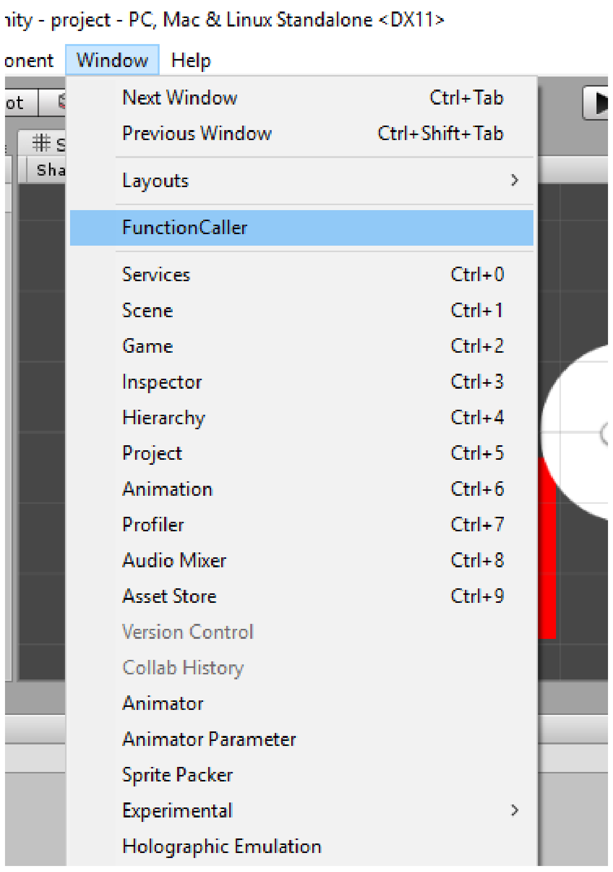
FunctionCaller

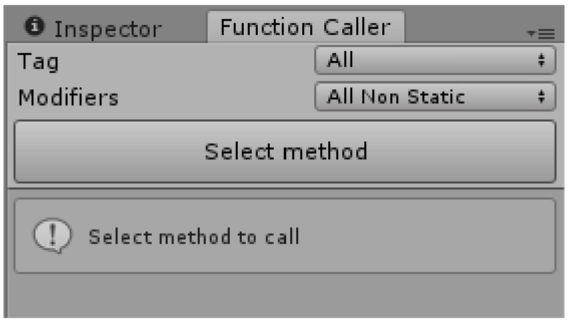
Basics

FunctionCaller allows you to call methods of any object on your scene while playing.

Open FunctionCaller window by selecting Window->FunctionCaller.



Window will appear near the inspector window. You can move it wherever you want, but pay attention that it is designed to be vertical-oriented.



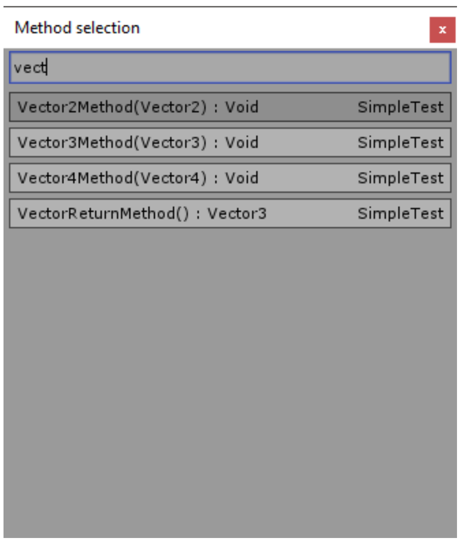
Tag: All
Modifiers: All Non Static

Select one or more objects on the scene. Now you can invoke methods of scripts, attached to object(s).

There are two ways of how to select a method:

1. Use search.

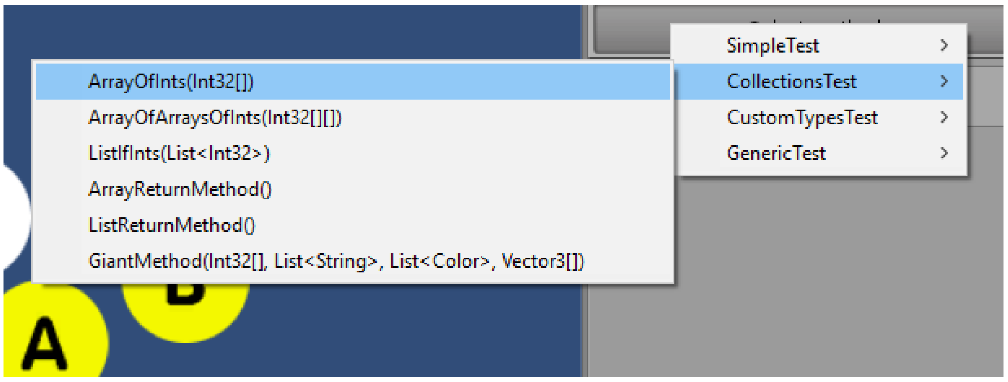
Click the “Select method” button and type in part of the method name:


Method selection
vect
Vector2Method(Vector2) : Void SimpleTest
Vector3Method(Vector3) : Void SimpleTest
Vector4Method(Vector4) : Void SimpleTest
VectorReturnMethod() : Vector3 SimpleTest

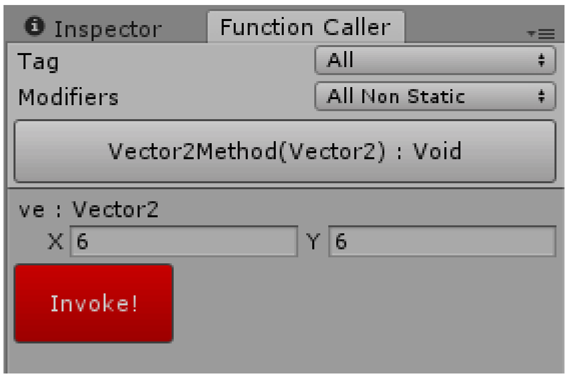
Select method from the list by using up and down buttons (and hit enter) or mouse.

2. Use context menu

The other way to select method is context menu. Right-click "Select method" button, context menu will appear. Then select desired component and method.



After you have selected the method, fill-in arguments and click red invoke button.



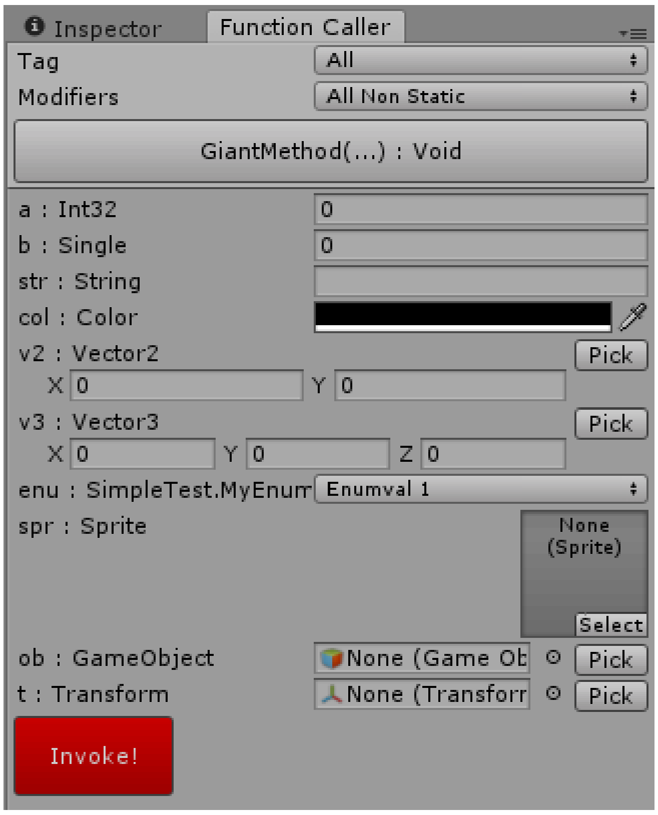
Arguments

There are different types of arguments and fields for them.

Simple types

Simple arguments are arguments that can be represented as one built-in Unity editor GUI field. For example, int, string, Vector3, Color, :Object (any subtype of UnityEngine.Object or itself).

Some of simple arguments are included in “Giant method” on demo scene (Ball_A->SimpleTest->GiantMethod).


Inspector | Function Caller

Tag: All

Modifiers: All Non Static

GiantMethod(...) : Void

a : Int32 — 0

b : Single — 0

str : String

col : Color

v2 : Vector2 — Pick — X 0 Y 0

v3 : Vector3 — Pick — X 0 Y 0 Z 0

enu : SimpleTest.MyEnum — Enumval 1

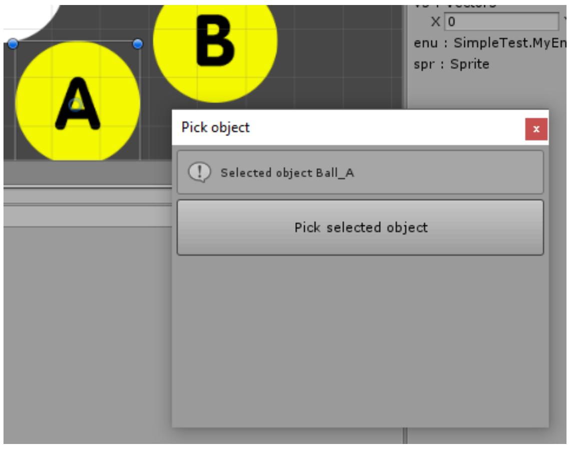
spr : Sprite — None (Sprite) — Select

ob : GameObject — None (Game Ob — Pick

t : Transform — None (Transforr — Pick

Invoke!

Values for fields that have “Pick” button may be selected directly on the scene view. Click the “Pick” button, window will appear:



X 0

enu : SimpleTest.MyEn

spr : Sprite

Pick object

Selected object Ball_A

Pick selected object

Now select desired object on the scene view and click “Pick select object” button.

For “Object” and “GameObject” types – selected game object will be set as value.

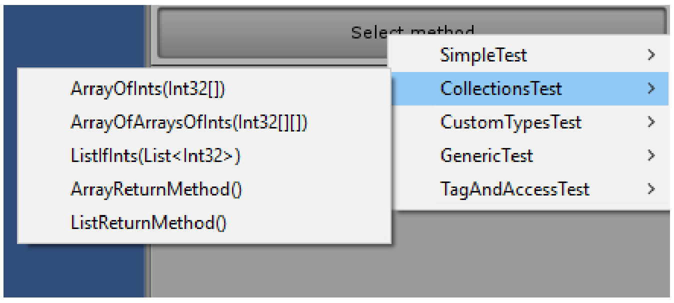
For “Transfrom”, “SpriteRenderer” and other components (or scripts) types – the component will be taken from the selected GameObject.

For “Vector2” and “Vector3” types – the position of selected game object will be taken as value.

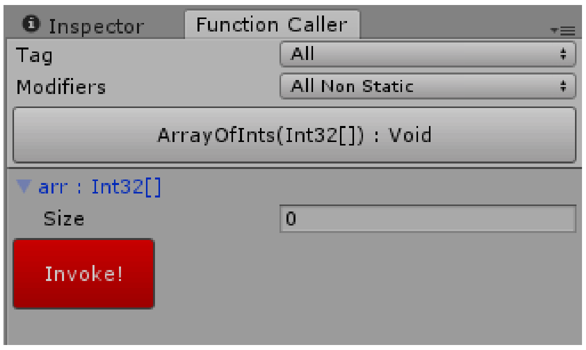
Collections

Function caller can create one dimensional arrays ([i, j] is not supported, but [ ][ ] / [ ][ ][ ] /... works) and lists of any type. For example, int[ ], float[ ][ ], List<Color>, CustomClass[ ] (information about custom classes will be given later).

Have a look at methods in “CollectionsTest”:

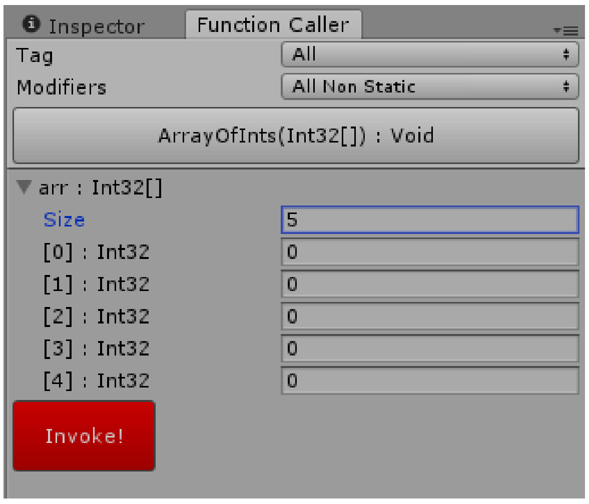


For array and list parameters you will be asked to enter size of collection:



Tag: All
Modifiers: All Non Static
ArrayOfInts(Int32[]) : Void
arr : Int32[]
Size: 0

Enter size and fields for values will appear:

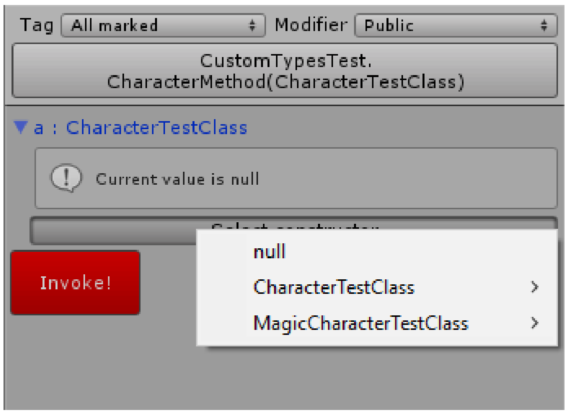


Tag: All
Modifiers: All Non Static
ArrayOfInts(Int32[]) : Void
arr : Int32[]
Size: 5
[0] : Int32: 0
[1] : Int32: 0
[2] : Int32: 0
[3] : Int32: 0
[4] : Int32: 0

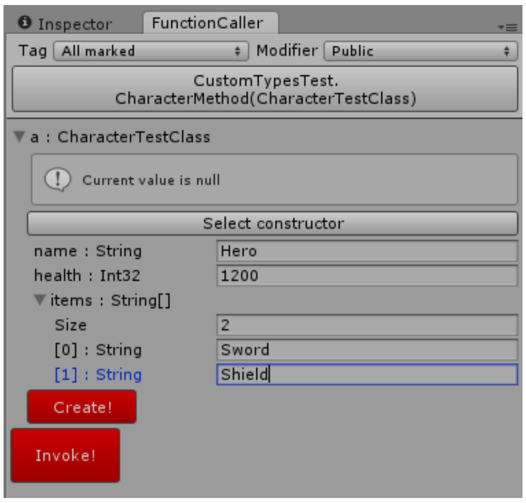
Now you can set each item value.

Custom types

Function caller allows you to create any objects by using their constructors. Constructor may have parameters, which you will be able to set the same way as method parameters. To begin creating object click “Select constructor”.

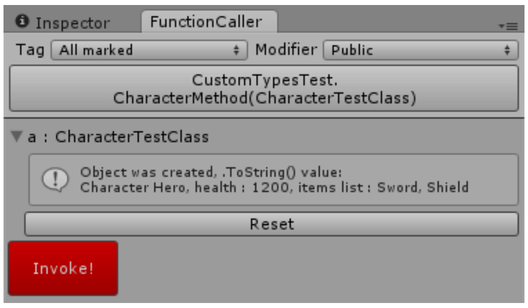


In this example, MagicCharacterTestClass extends CharacterTestClass, so you can create objects of both types for this parameter. Or you can set value as null, if you want.
Once you've selected constructor, you will be able to set values:



name : String Hero
health : Int32 1200
items : String[]
Size 2
[0] : String Sword
[1] : String Shield

After you have entered all values, click "Create!" button.



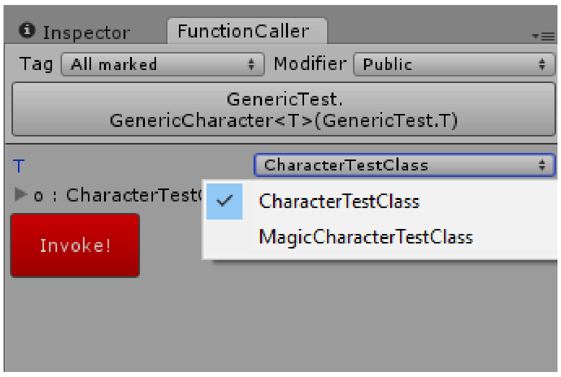
Object was created, .ToString() value:
Character Hero, health : 1200, items list : Sword, Shield

Object is created and its ToString() value is displayed. You can reset it to null by click "Reset" and then select new constructor to create object again.

Important! Check that you have created (clicked create button) objects for all parameters of custom types. Otherwise, nulls will be passed as parameters.

Generic methods

Function caller supports generic methods. Once you have selected such method, you will be able to select generic parameters:



After that generic parameters will change type.

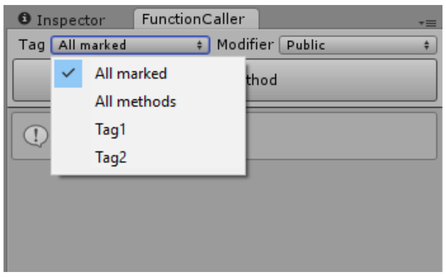
Tags, marks and modifiers

By default, function caller shows all non-static public methods. You can change these settings.

Tags and marks

Method is marked if it has [CallableFunction] attribute.

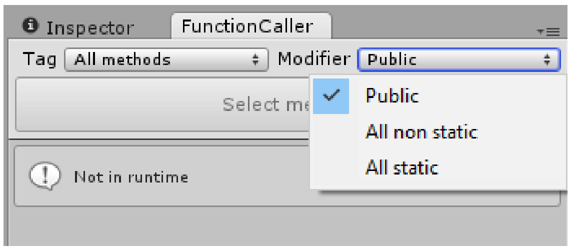
You may also set tags to methods via passing them to CallableFunction attribute constructor: `[CallableFunction("Tag1", "Tag2")]. This may be useful, if you have a lot of classes and methods and need to use only part of them (or group them by tag).`



You can select only one tag. All methods that are marked with this tag will be displayed. Method with attribute given above will be displayed by both tags: “Tag1” and “Tag2”.
If you select “All marked” all marked methods will be displayed.

Modifiers

There are three modes for modifiers.

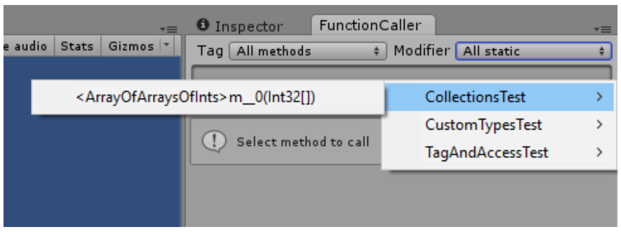


Public
Only non-static public methods will be displayed.

All non-static
All non-static methods will be displayed (public, private, protected, internal).

All static
All static methods will be displayed (public, private, protected, internal).

Note: some non-public methods may look strange; it may be lambdas in LINQ queries or anonymous delegates that are compiled to separate static methods.

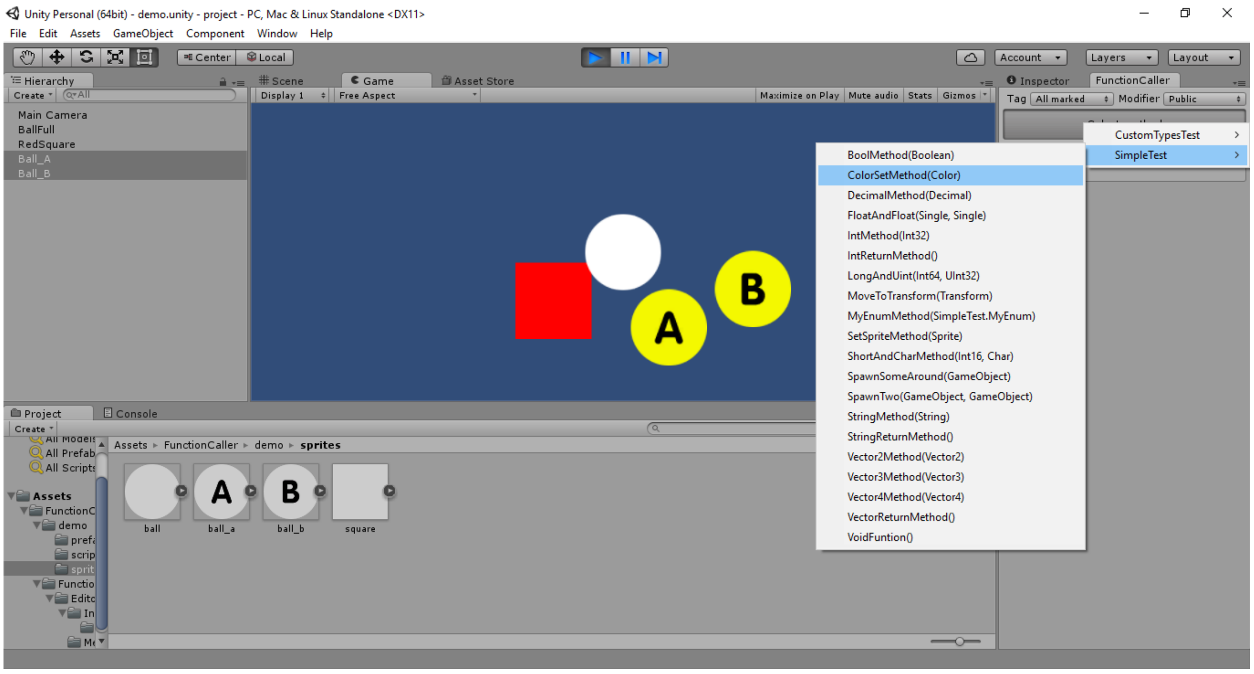


Multi-object invocation

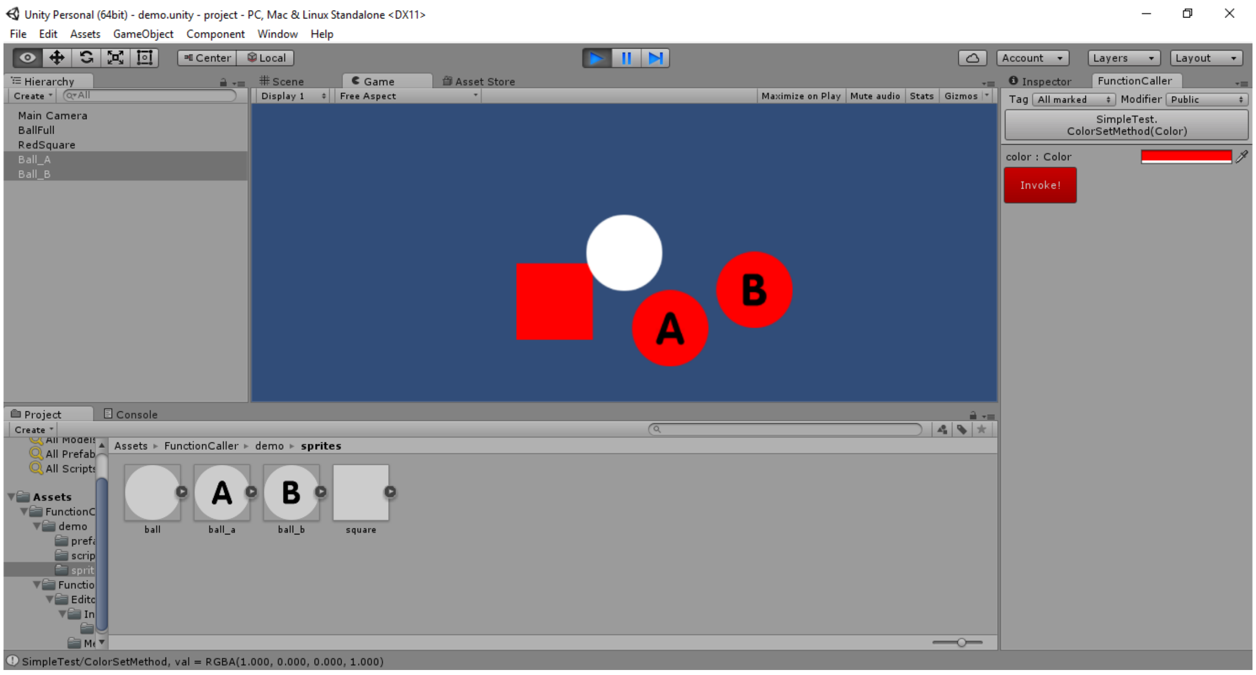
Function caller allows you to invoke method of 2 or more objects.

Select 2 or more objects on scene or hierarchy while playing. Next, click "Select method" button. You will be able to call methods of components that are added to all selected objects.

Ball_A components	Ball_B components	Available for invoke
SimpleTest	SimpleTest	+ // both have SimpleTest
CollectionsTest	-	- //Ball_B hasn't CollectionTest
CustomTypesTest	CustomTypesTest	+
GenericTest	-	-
-	TagAndAccessTest	-


Unity Personal (64bit) - demo.unity - project - PC, Mac & Linux Standalone <DX11>
File Edit Assets GameObject Component Window Help
Center Local
Account Layers Layout
Hierarchy
Scene Game Asset Store
Display 1 Free Aspect
Maximize on Play Mute audio Stats Gizmos
Inspector FunctionCaller
Tag All marked Modifier Public
Main Camera
BallFull
RedSquare
Ball_A
Ball_B
CustomTypesTest
SimpleTest
BoolMethod(Boolean)
ColorSetMethod(Color)
DecimalMethod(Decimal)
FloatAndFloat(Single, Single)
IntMethod(Int32)
IntReturnMethod()
LongAndUint(Int64, UInt32)
MoveToTransform(Transform)
MyEnumMethod(SimpleTest.MyEnum)
SetSpriteMethod(Sprite)
ShortAndCharMethod(Int16, Char)
SpawnSomeAround(GameObject)
SpawnTwo(GameObject, GameObject)
StringMethod(String)
StringReturnMethod()
Vector2Method(Vector2)
Vector3Method(Vector3)
Vector4Method(Vector4)
VectorReturnMethod()
VoidFuntion()
B
A
Project Console
Assets
Assets ▸ FunctionCaller ▸ demo ▸ sprites
ball ball_a ball_b square

In this case we can invoke methods of two components. For example, select "ColorSetMethod(Color)" from "SimpleTest" component.


Unity Personal (64bit) - demo.unity - project - PC, Mac & Linux Standalone <DX11>
File Edit Assets GameObject Component Window Help
Center Local
Account Layers Layout
Hierarchy
Scene Game Asset Store
Display 1 Free Aspect
Maximize on Play Mute audio Stats Gizmos
Inspector FunctionCaller
Main Camera
BallFull
RedSquare
Ball_A
Ball_B
A
B
Project Console
Assets ▸ FunctionCaller ▸ demo ▸ sprites
ball ball_a ball_b square
Invoke!

Tag: All marked
Modifier: Public
SimpleTest.ColorSetMethod(Color)
color : Color

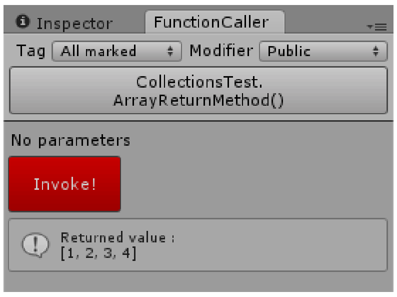
```
SimpleTest/ColorSetMethod, val = RGBA(1.000, 0.000, 0.000, 1.000)
```

When “Invoke!” button is clicked method is invoked for all selected objects (color of Ball_A and Ball_B changes).

Returned values

There is an ability in FunctionCaller to see returned values. Once you invoke method, returned value(s) will be displayed below.

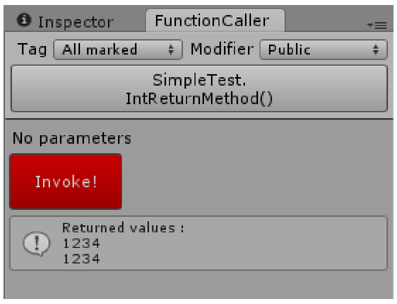
For single object invocation (method returns int[ ]):



Tag: All marked
Modifier: Public
CollectionsTest.ArrayReturnMethod()
No parameters

```
Returned value :
[1, 2, 3, 4]
```

Form multiple objects invocation (2 objects, method returns int):



Tag: All marked
Modifier: Public
SimpleTest.IntReturnMethod()
No parameters

```
Returned values :
1234
1234
```

(Returned values are displayed in separate lines)

List of supported types

Type	Field for given type
bool	BoolParameterInputField
char	CharParameterInputField
Color	ColorParameterInputFiled
decimal	DecimalParameterInputField
string	StringParameterInputField
float	FloatParameterInputField
double	DoubleParameterInputField
Rect	RectParameterInputField
long, int, uint, short, ushort, byte, sbyte	IntParameterInputField
AnimationCurve	AnimationCurveInputField
Vector(2,3,4)	VectorParameterInputField
Array	ArrayParameterInputField
List<T>	ListParameterInputField
*:UnityEngine.Object	ObjectParameterInputField
enum	EnumParameterInputField
Custom class or interface	CustomTypeInputField