# Observables

**Tie any serializable object to a UnityEvent.**

**Jacob Homanics**

Version 1.1.0

Observables use an event based approach to development using UnityEvents. They allow the developer to set up their game in-editor in a decoupled manner. Any serializable object in Unity can utilize this functionality with great power. Take advantage of performance gains by ony accessing values when they are changed as opposed to every Update frame. Improve code quality by naturally being decoupled with the system.

## Features

- Store a reference or value to a script instance.
- Set a reference or value on a script instance which fires a UnityEvent.

# Basic API

## Abstract Observable

The MonoBehaviour instance which holds the object of which a UnityEvent is attached to. The UnityEvent will fire any time the object has been set.

The base implementation does not use generics and does not yet implement a UnityEvent. However, it still does contain information regarding the object in a general sense in the form of a c# type: object.

### **Methods**:

- **Abstract object GetValue(void)**: returns the observable's object as a c# object. Complete implementation done in the generic Observable class.
- **Abstract void SetValue(object)**: Sets the observable's object value to the passed in parameter. Complete implementation done in the generic Observable class.

## Abstract Observable<ObservableType, ValueType, EventType>

The generic implementation provides a lot more information regarding what is going on with the Observable. You get the actual type of the observable, the type of the object which belongs to the Observable, and the type of the event attached to the observable.

### **Properties**:

- **ValueType Value {get; set;}**: The value being stored by the observable. The UnityEvent associated with the observable will fire from within this Set method.

### **Fields**:

- **EventType ValueChanged**: Must derive from ValueChanged<ObservableType> where ObservableType : Observable. The event which will fire when the Value variable is set.
- **Bool fireOnStart**: Determines whether the ValueChanged event will be invoked in the Start method.

### **Methods**:

- **Void SetValue(object obj):** Casts the passed in object to ValueType and sets the Value variable to the casted type.
- **object GetValue(void)**: Returns the Value variable as an object.
- **Void SetValueWithoutEvent(ValueType)**: Sets the Value variable without invoking the event attached to it.

## ValueChanged<ObservableType>

The base class of the event that will fire when its observable owner's Value variable has been set. The generic implementation for the ValueChanged UnityEvent. A new derivation will need to be made to accommodate each new Observable class type that is created.

## ObservableFloat

Inherits from Observable<ObservableFloat, float, FloatChanged>. Allows a float variable to be observed and fires an event when it is set.

**Methods**:

- **Void Add(float value):** Adds the passed in parameter to the Value.
- **Void Subtract(float value)**: Subtracts the passed in parameter from the Value.

## FloatChanged

Inherit from ValueChanged<ObservableFloat>. To be used as the EventType generic parameter in ObservableFloat.

## ObservableBool

Inherits from Observable<ObservableBool, bool, BoolChanged>. Allows a bool variable to be observed and fires an event when it is set.

## BoolChanged

Inherit from ValueChanged<ObservableBool>. To be used as the EventType generic parameter in ObservableBool.

## ObservableTransform

Inherits from Observable<ObservableTransform, Transform, TransformChanged>. Allows a Transform variable to be observed and fires an event when it is set.

# TransformChanged

Inherit from ValueChanged<ObservableTransform>. To be used as the EventType generic parameter in ObservableTransform.

# ObservableValueDisplayer

A UI based class which will display the Value of an observable in a Text object.

**Fields**:

- **Text text**: Text object used to display the Value of an Observable.

**Methods**:

- **Void Set(Observable):** Converts the Value of the passed in Observable to a string then sets the Text object's text value to the string.

# Report a Bug

You may send all bugs to the email:
homanicsjake@gmail.com
You can send whatever you believe is necessary for me to examine and solve the bug. This may include projects, scripts, screenshots, videos.

# Request a feature

I have set it up to be easily expanded upon where you can easily implement features. If you believe your own implemented feature or you think a feature fits the package, then send any information regarding that to:
homanicsjake@gmail.com

# Contact Me

I believe this package has a lot of potential. I see it tried and proven in my own projects. I am open to all opportunities and ideas regarding this project. Potential collaborators, criticizers, future opportunists, celebrators, are all welcome to email me at: homanicsjake@gmail.com.