Earth Engine User Interface Coding

Earth Engine UI Coding

Sufyan Abbasi
sufy@google.com

Goals for this session:

1. Learn about the various Earth Engine widgets and how they operate.

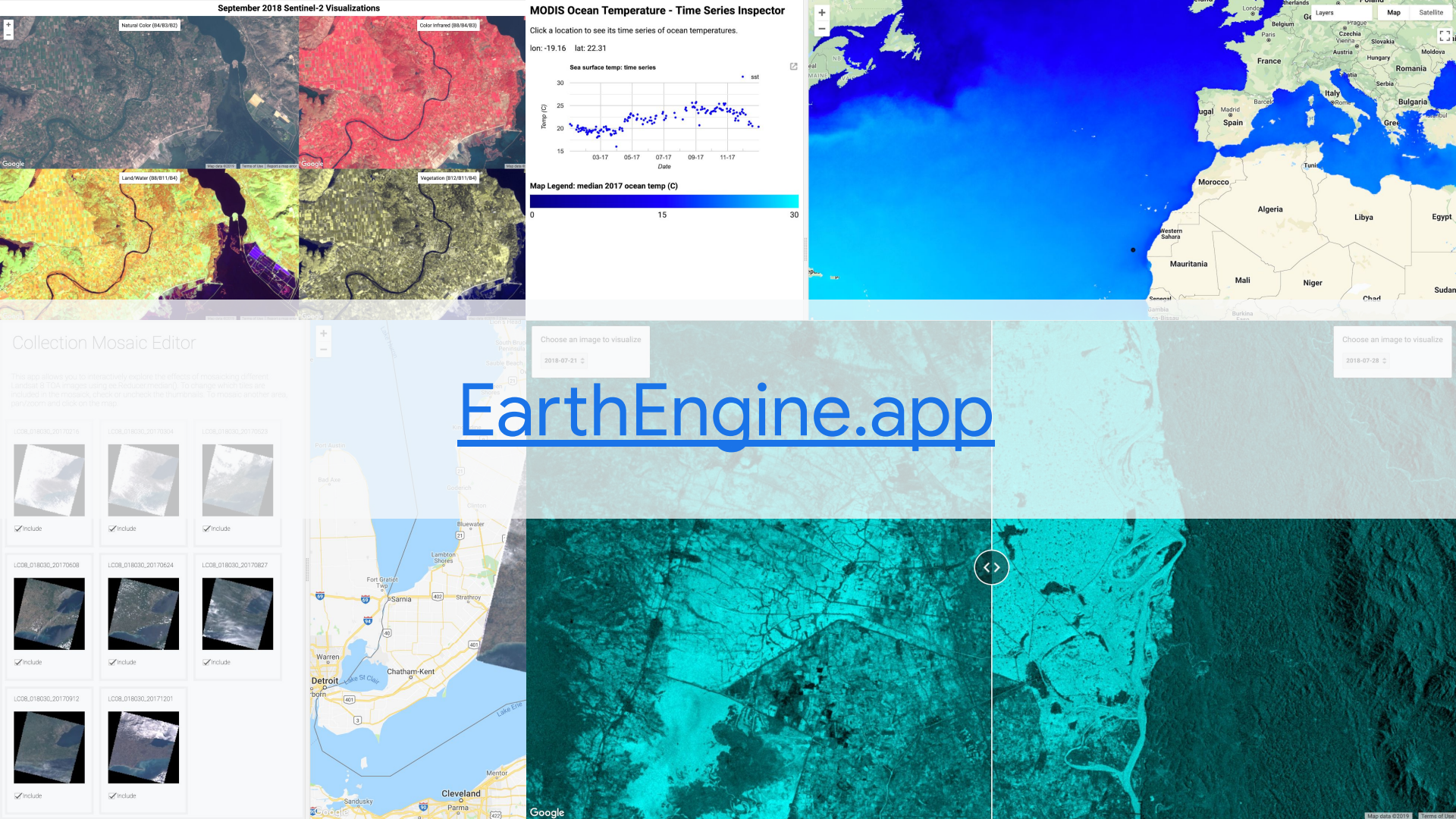
2. Learn techniques to structure your code to make developing UIs easier.

Logistics

1. View this slidedeck here: tinyurl.com/g4g-ui-coding

2. Get the examples repository here:
   https://code.earthengine.google.com/?accept_repo=users/sufy/g4g-ui-coding

UI Coding in Earth Engine


September 2018 Sentinel-2 Visualizations
Natural Color (B4/B3/B2)
Color Infrared (B8/B4/B3)
Land/Water (B8/B11/B4)
Vegetation (B12/B11/B4)
MODIS Ocean Temperature - Time Series Inspector
Click a location to see its time series of ocean temperatures.
lon: -19.16 lat: 22.31
Sea surface temp: time series
sst
Temp (C)
Date
Map Legend: median 2017 ocean temp (C)
0
15
30
Collection Mosaic Editor
Choose an image to visualize
2018-07-21
2018-07-28
EarthEngine.app

What kind of Widgets do we have in Earth Engine?

1. Widgets that let you **display** information.

2. Widgets that let users **input parameters** or interact with your script.

3. Widgets that let you **layout** other widgets.

What kind of Widgets do we have in Earth Engine?

1. Widgets that let you **display** information.

2. Widgets that let users input parameters or interact with your script.

3. Widgets that let you layout other widgets.

Display Widget: ui.Label

ui.Label

ui.Chart

ui.Thumbnail

```
var label = ui.Label("Hello!");
label.setValue("Changed label.");
label.setUrl("https://earthengine.google.com");
Map.add(label);
```

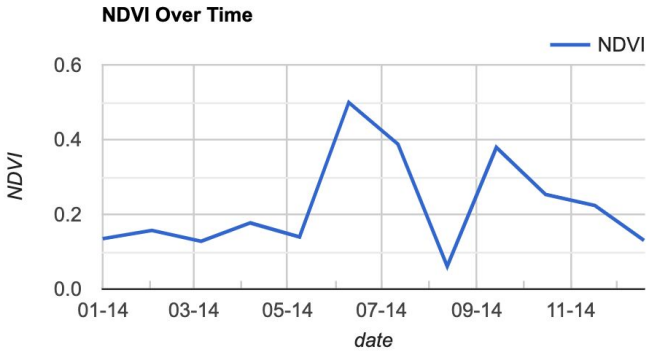
Display Widget: ui.Chart

ui.Label

ui.Chart

ui.Thumbnail

```
var chart = ui.Chart.image.series(
    ndvi, point, ee.Reducer.mean(), 200);
  chart.setOptions({
    title: 'NDVI Over Time',
    vAxis: {title: 'NDVI'},
    hAxis: {
        title: 'date',
        format: 'MM-yy',
        gridlines: {count: 7},
    },
  });
Map.add(chart);
```



Display Widget: ui.Thumbnail

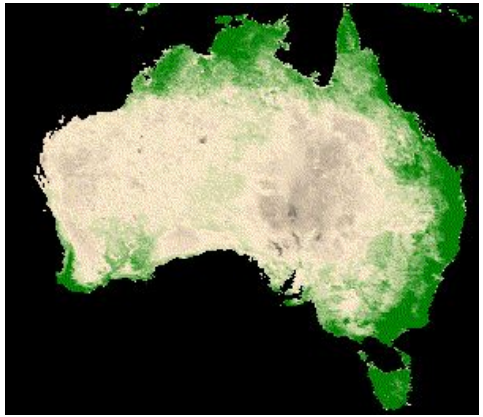
See: Animated Thumbnail Example

ui.Label

ui.Chart

ui.Thumbnail

```
ui.Thumbnail({
  image: collection,
  params: {
      crs: 'EPSG:3857',
      dimensions: '300',
      region: rect,
      min: -2000,
      max: 10000,
      palette: 'black, blanchedalmond, green, green',
      framesPerSecond: 12,
    }
});
```



What kind of Widgets do we have in Earth Engine?

1. Widgets that let you display information.

2. Widgets that let users **input parameters** or interact with your script.

3. Widgets that let you layout other widgets.

Input Widget: ui.Buttons

ui.Button

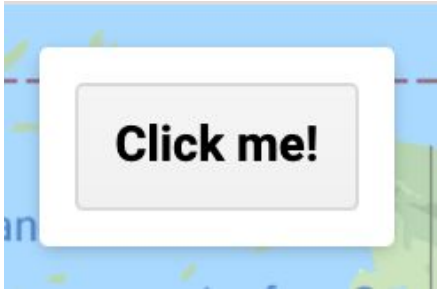
ui.CheckBox

ui.DateSlider

ui.Select

ui.Textbox

```
var button = ui.Button("Click me!");
button.onClick(function() {
  print("I was clicked");
});
Map.add(button);
```



Input Widget: ui.Checkbox

ui.Button

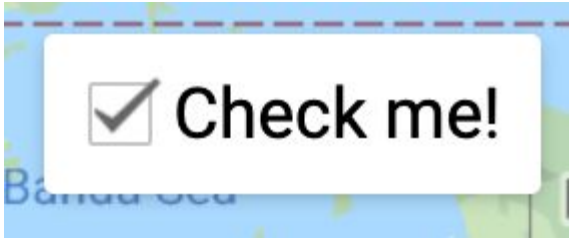
ui.CheckBox

ui.DateSlider

ui.Select

ui.Textbox

```
var checkbox = ui.Checkbox("Check me!");
checkbox.onChange(function(isChecked) {
  if (isChecked) {
    print("I'm checked");
  } else {
    print("I'm unchecked");
  }
});
Map.add(checkbox);
```



Input Widget: ui.DateSlider

ui.Button

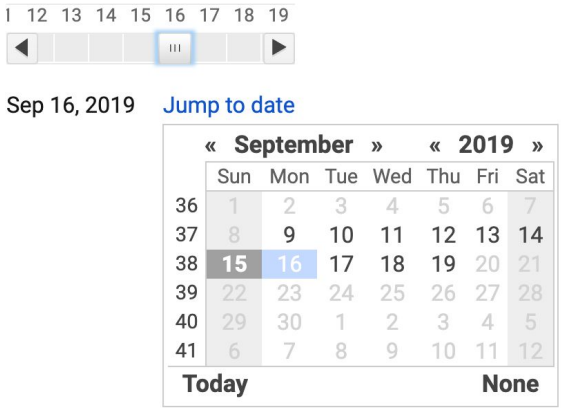
ui.CheckBox

ui.DateSlider

ui.Select

ui.Textbox

```
var dateSlider = ui.DateSlider({
  start: "2019-09-10",
  end: "2019-09-20",
  value: "2019-09-16",
});
dateSlider.onChange(function(dateRange) {
  print(dateRange);
});
Map.add(dateSlider);
```



Input Widget: ui.Select

ui.Button

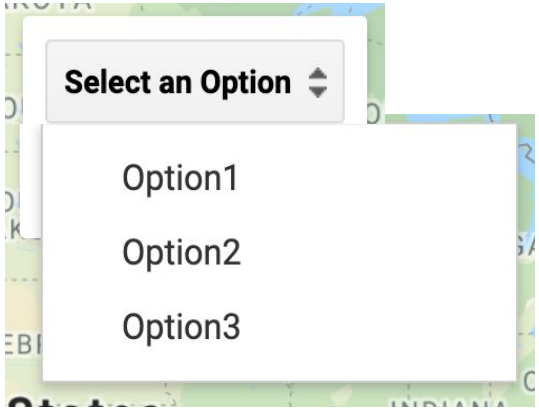
ui.CheckBox

ui.DateSlider

ui.Select

ui.Textbox

```
var select = ui.Select(
  ['Option1', 'Option2', 'Option3'],
  "Select an Option");
select.onChange(function(selected) {
  print("Selected Option: " + selected);
});
Map.add(select);
```



Input Widget: ui.Textbox

ui.Button

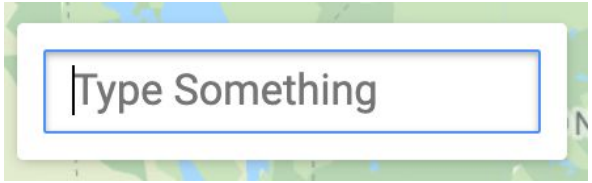
ui.CheckBox

ui.DateSlider

ui.Select

ui.Textbox

```
var textbox = ui.Textbox("Type
Something");
textbox.onChange(function(text) {
  print("Typed: " + text);
});
print(textbox.getValue());
Map.add(textbox);
```



What kind of Widgets do we have in Earth Engine?

1. Widgets that let you display information.

2. Widgets that let users input parameters or interact with your script.

3. Widgets that let you **layout** other widgets.

Layout Widget: ui.Map

ui.Map

ui.Panel

ui.SplitPanel

```
var map = ui.Map();

ui.root.clear();
ui.root.add(map);
map.addLayer(...);
map.onClick(...);
```

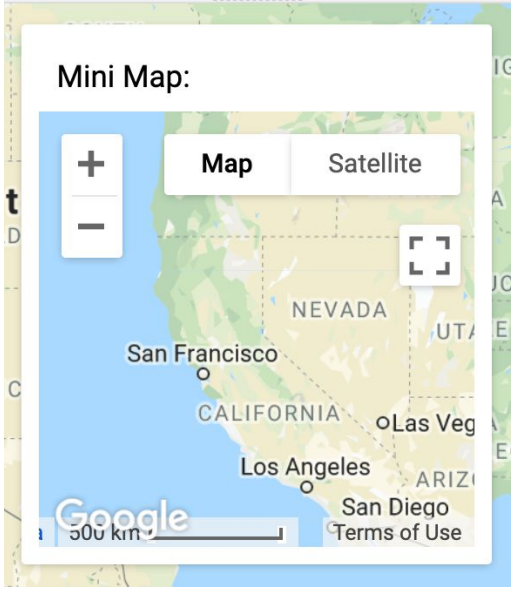
Layout Widget: ui.Panel

ui.Map

ui.Panel

ui.SplitPanel

```
var panel = ui.Panel();
var title = ui.Label("Mini Map:");
panel.add(title);
panel.add(ui.Map());
panel.setLayout(
    ui.Panel.Layout.flow("vertical"))
Map.add(panel);
```



Layout Widget: ui.SplitPanel

ui.Map

ui.Panel

ui.SplitPanel

```
var leftMap = ui.Map();
var rightMap = ui.Map();

var splitPanel = ui.SplitPanel({
  firstPanel: leftMap,
  secondPanel: rightMap,
  wipe: true, // false for resizable panels
  orientation: "horizontal",
});
ui.root.clear();
ui.root.add(splitPanel);
```

Core JavaScript Concepts

Functions

A function is a block of code that executes when you call it by its name. They may also accept some inputs:

```
function sayHello(name) {

    return "Hello, my name is " + name;

}

print(sayHello("Sufyan"));
```

Console Output:

```
> Hello, my name is Sufyan
```

Dictionaries

An object is a data structure that allows you store key-value pairs. Input a key and it returns the value that you passed in:

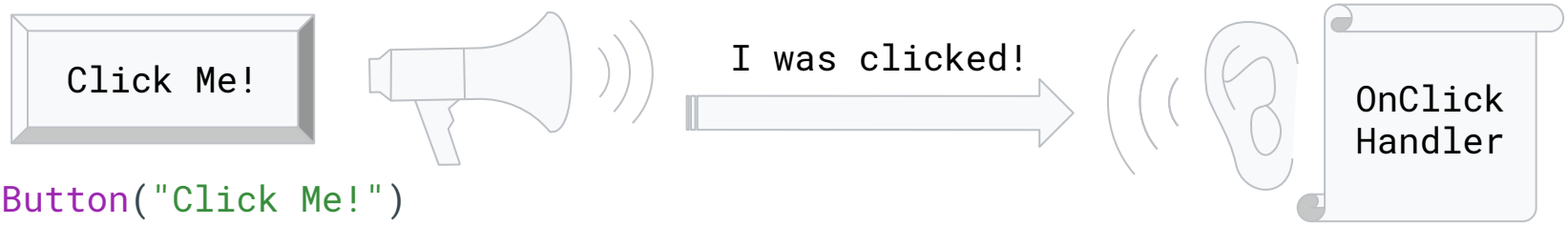
```
var imageCollections = {
    landsat: 'LANDSAT/LC08/C01/T1_RT',
    sentinel: 'COPERNICUS/S5P/NRTI/L3_NO2',
}
print(imageCollections.landsat)
print(imageCollections['sentinel'])
imageCollections.srtm = 'CGIAR/SRTM90_V4'
```

Console Output:

```
> LANDSAT/LC08/C01/T1_RT

> COPERNICUS/S5P/NRTI/L3_NO2
```

Event Handling: making your widgets interactive

By binding an event handler, or **callback**, to a user event, you can execute a function when the user does something.




```
ui.Button("Click Me!")
```


```
button.onClick(function() {
  button.setLabel("Clicked.");
})
```

Event Handling Patterns

- All of our widgets (except for ui.Label) have one of two event handlers: **onClick** or **onChange**, depending on the widget.

- All of the **onChange** callback functions share the same pattern for inputs: the first parameter is the new input value and the second is the widget itself:

```
textbox.onChange(function(value, widget) {
    print("Widget: " + widget + " has value: " + value);
});
```

Working with Widgets

Widget Workflow

1. Make a widget:

```
var button = ui.Button("Click Me");
```

2. Make it listen to events:

```
button.onClick(function() {
   print("I was clicked.");
});
```

3. Add it to another widget:

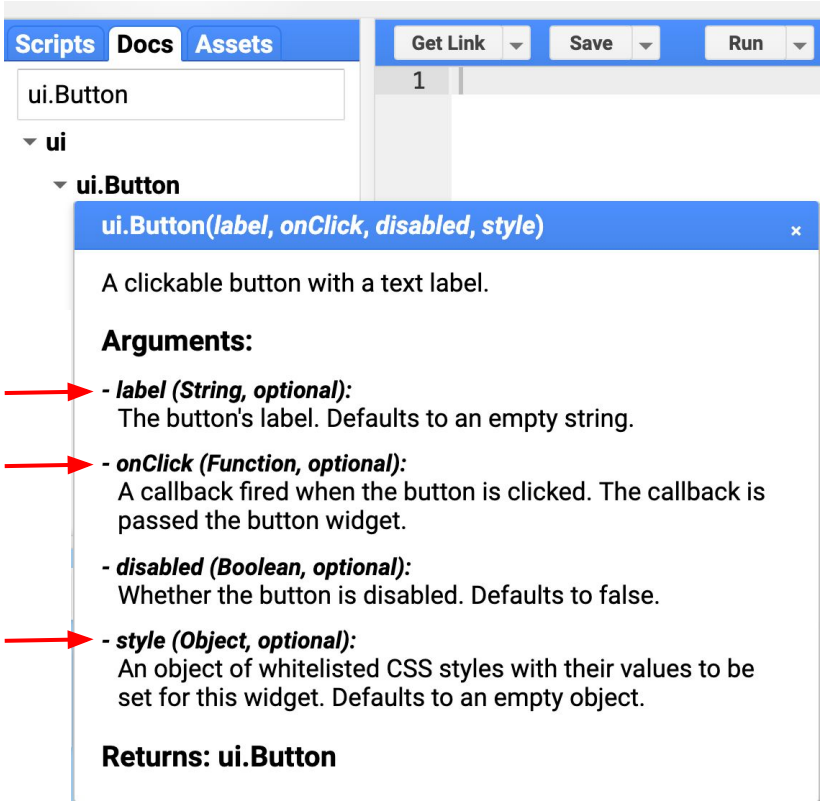
```
Map.add(button);
```

4. Style it:

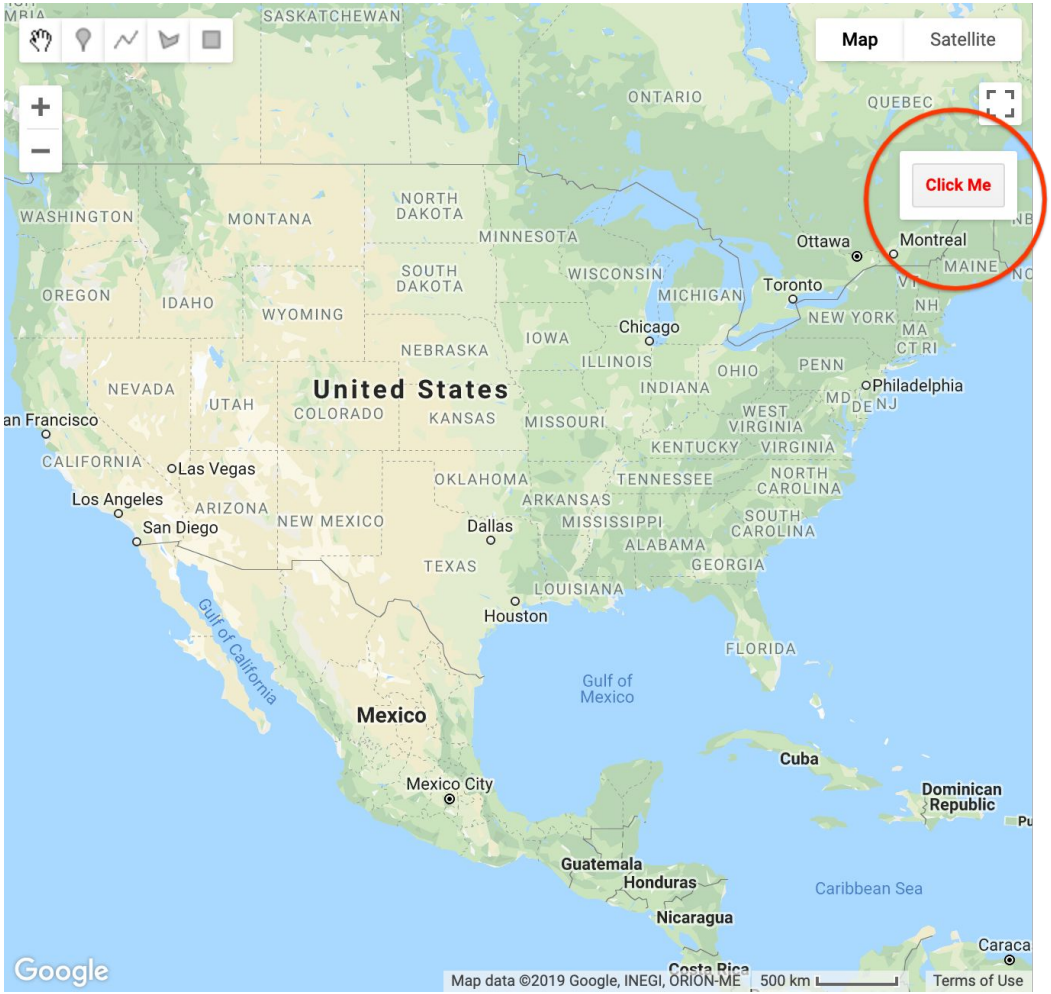
```
button.style().set('color', 'red');

button.style().set('position', 'top-right');
```

A more compact form:

```
var button = ui.Button({
    label: "Click Me",
    onClick: function() {
        print("I was clicked.");
    },
    style: {
        color: 'red',
        position: 'top-right',
    },
});
Map.add(button);
```



Result:

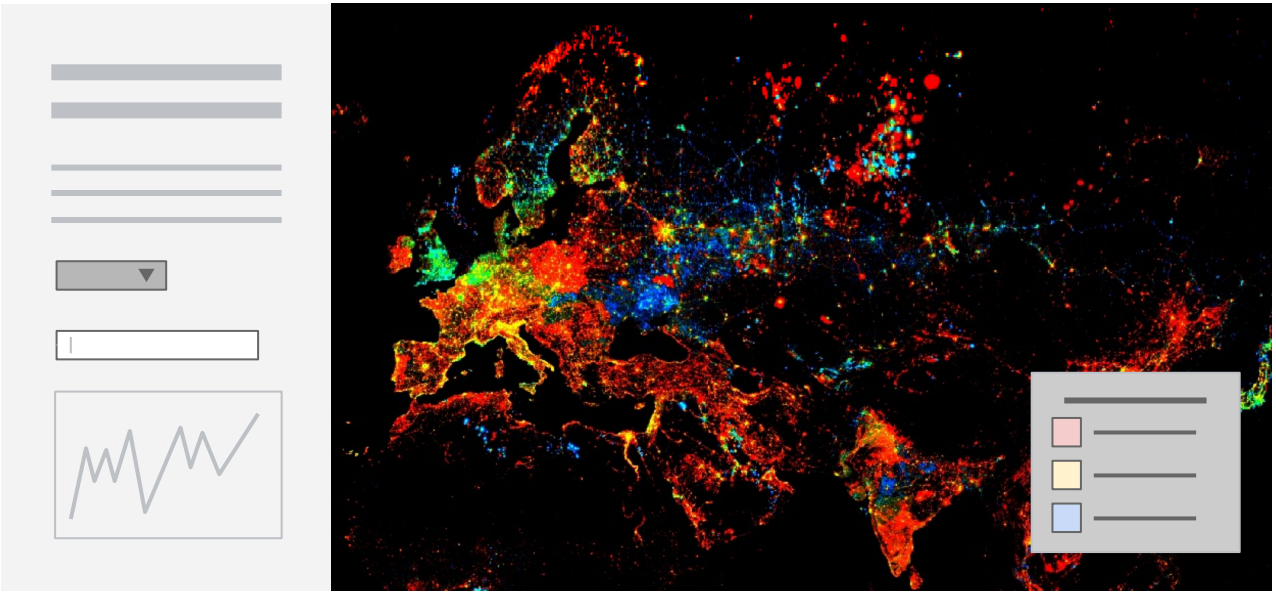


#GeoForGood19

Layouts

Making Layouts with ui.Panel and ui.Map

EE Apps “Bread and Butter” Template:



Making Layouts with ui.Panel and ui.Map

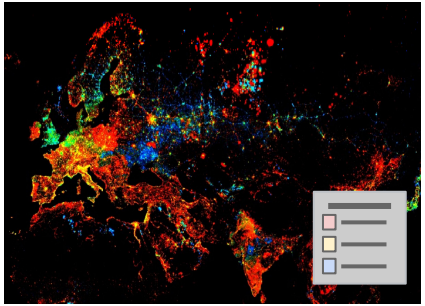
Step 1: Make the side panel.

```
var sidePanel = ui.Panel({
  layout: ui.Panel.Layout.flow('vertical'),
  style: {
    height: '100%',
    width: '30%',
  },
})
```

Making Layouts with ui.Panel and ui.Map

Step 2: Make the map and legend.

```
var map = ui.Map();
var legendPanel = ui.Panel({
  widgets: [ui.Label("Legend")],
  layout: ui.Panel.Layout.flow('vertical'),
  style: {
    position: 'bottom-right',
  },
});
map.add(legendPanel);
```



Making Layouts with ui.Panel and ui.Map

Step 3: Combine the side panel and map into a ui.SplitPanel:

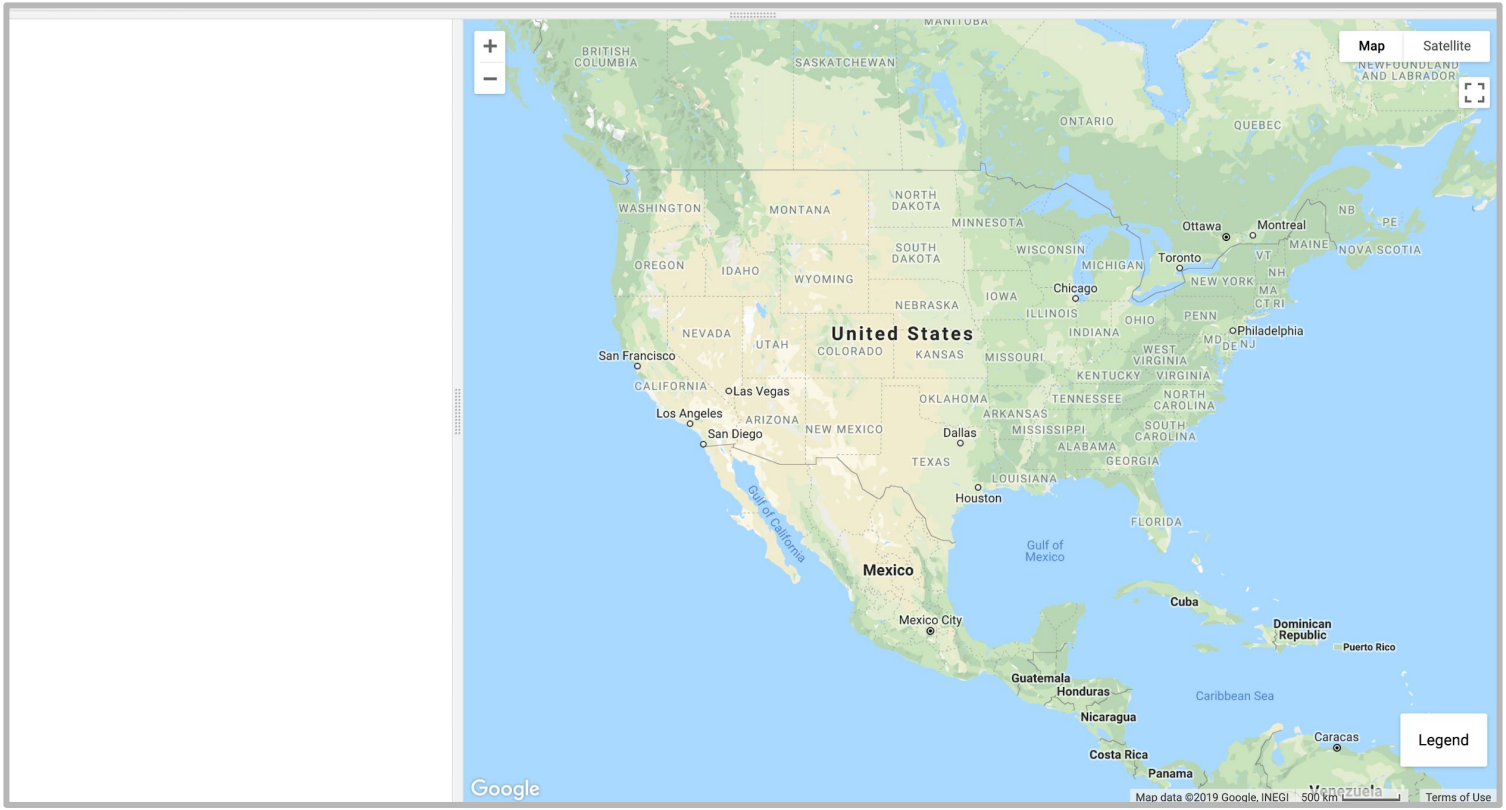
```
var splitPanel = ui.SplitPanel({

  firstPanel: sidePanel,

  secondPanel: map,
});
```

Making Layouts with ui.Panel and ui.Map

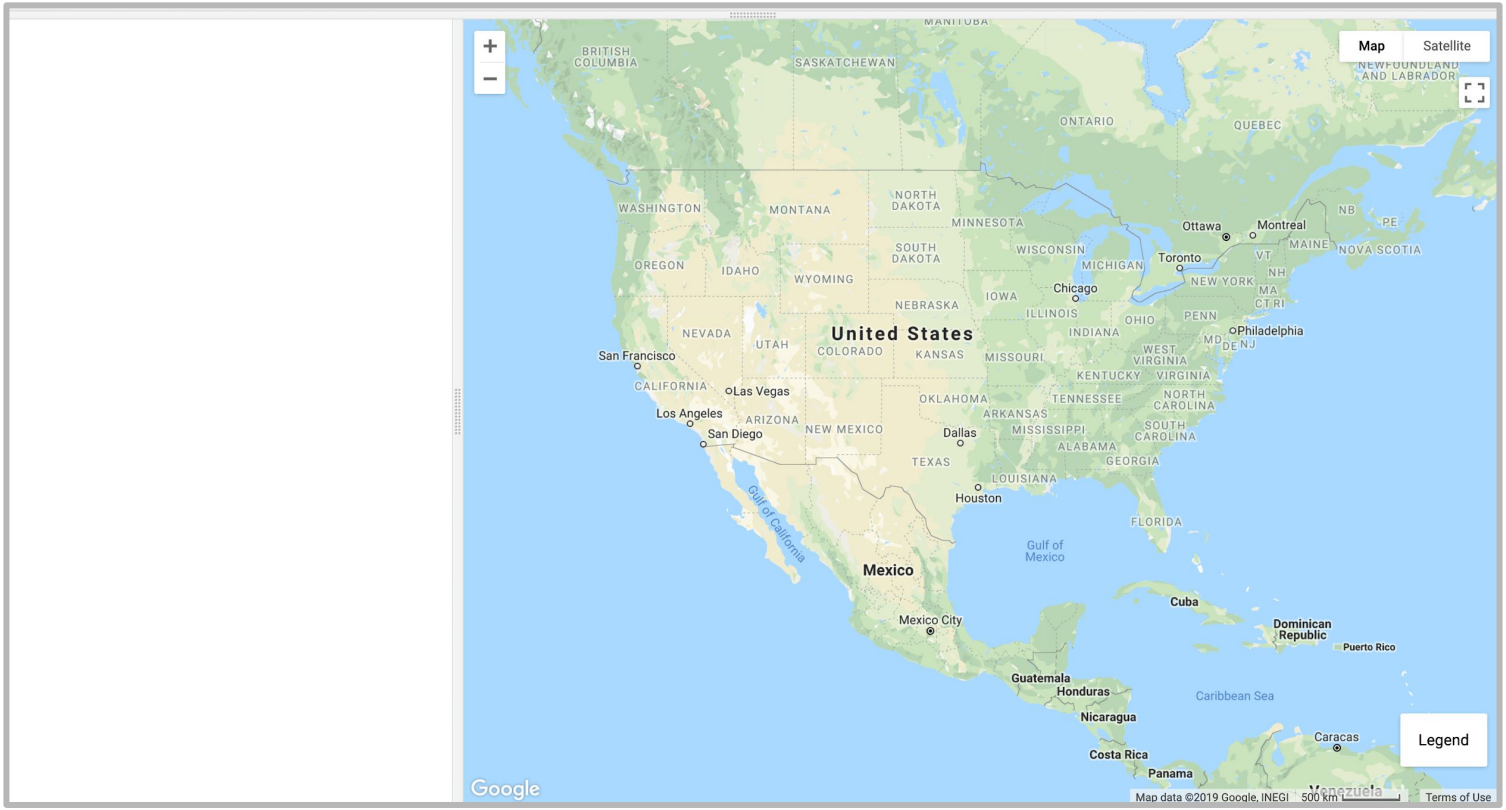
Step 4: Clear the ui.root and add the split panel:

```
ui.root.clear();
ui.root.add(splitPanel);
```

All that's left...



All that’s left… is the hard part.



Example: Widget Playground (see Logistics)

Widget Playground

This is a simple app to play around with widgets!

Click me!

Check me!

initial

12 13 14 15 16 17 18

Sep 15, 2019 Jump to date

Widget Best Practices

Use functions to make widgets

For each large widget component, make a function to produce it:

```
function makeTitlePanel(title, description) {
  title = ui.Label(title);
  description = ui.Label(description);
  return ui.Panel([title, description]);
}

var titlePanel = makeTitlePanel('My Cool App',
                                'This app was created in one day.');
Map.add(titlePanel);
```

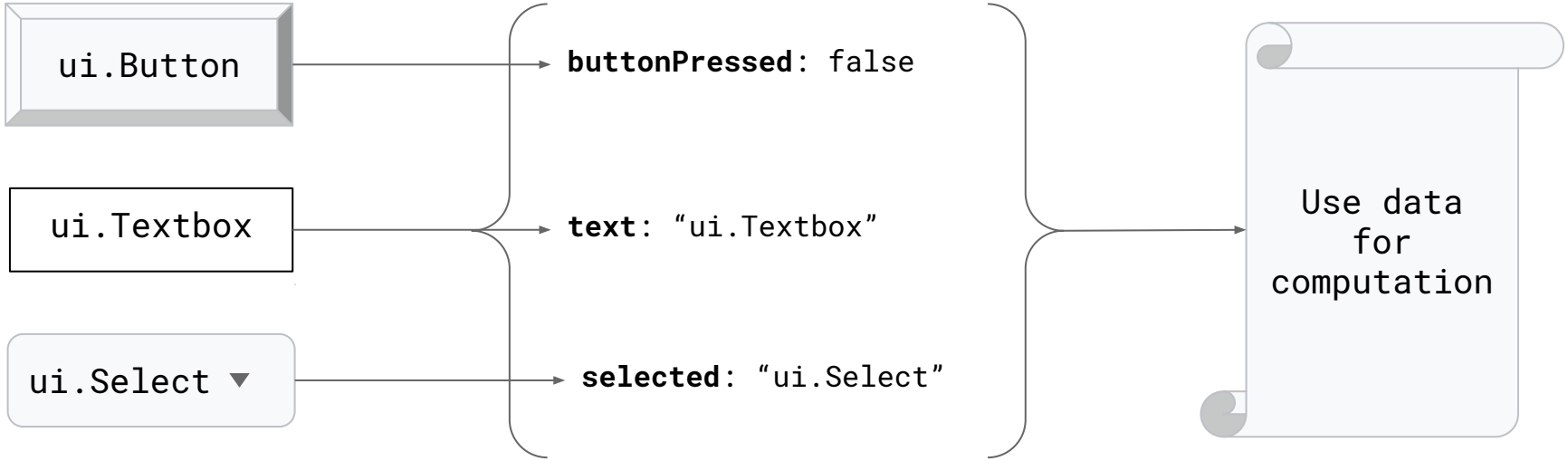
Consolidate your widget code

Put all your widget initialization in a function called `init()`, then call it at the end of the script. Remove as many global variables as you can:

```
function init() {
  var map = ui.Map();
  var sidePanel = ui.Panel();

  ui.root.clear();
  ui.root.add(sidePanel);
}

// End of script:
init();
```

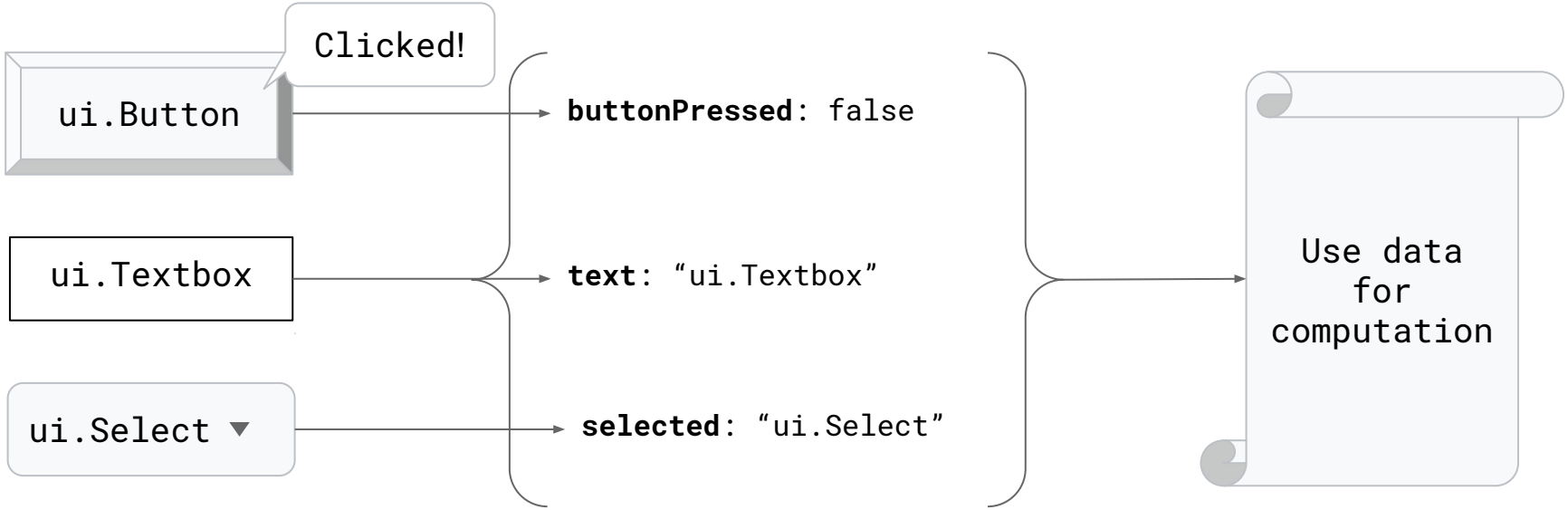
Consolidate the widget data

Use a dictionary to consolidate the data being displayed and produced by the widgets. The initial state of the widgets is also determined by the dictionary.



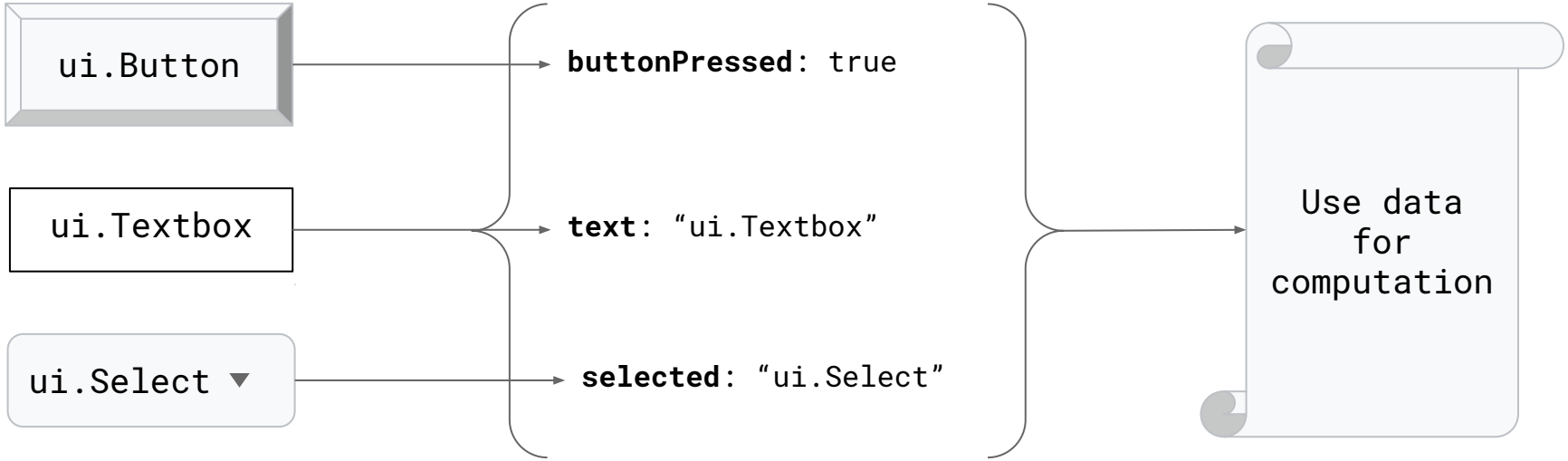
Consolidate the widget data

Use a dictionary to consolidate the data being displayed and produced by the widgets. The initial state of the widgets is also determined by the dictionary.



Consolidate the widget data

Use a dictionary to consolidate the data being displayed and produced by the widgets. The initial state of the widgets is also determined by the dictionary.



Styling Widgets

You can style widgets two ways:

1. Supply a **`style`** property in the widget constructor (emulates CSS):

```
style: {
    height: '100%',
    width: '30%',
    position: 'bottom-right'
}
```

2. Explicitly set a style property using **`widget.style().set(...)`**:

```
widget.style().set('position': 'bottom-right');
```

Styling Widgets

Easy styling properties (they use CSS syntax)

Property	Function	Example Values
padding	Puts padding around the widget	`'10px' '10px 0px' '10px 5px 0px 5px'`
margin	Adds space between widgets	`'10px' '10px 0px' '10px 5px 0px 5px'`
color	Changes the font color	`'white' 'rgb(255,255,255)' '#ffffff'`
backgroundColor	Changes the background color	`'white' 'rgb(255,255,255)' '#ffffff'`
fontWeight	Changes font weight	`'bold' 'lighter' '100'`