

WEB COMPONENTS FOR NON-IDEALISTS

WHAT ARE WEB
COMPONENTS?

BASICALLY: WIDGETS

Called "components"

```
<fancy-card>  
  <h3>Fridolin Frontend</h3>  
  <p>Frontend developer</p>  
</fancy-card>
```

INDEPENDENT, IMPORTABLE

```
<link rel="import"  
      href="components/fancy-card.html">
```

PART OF THE HTML STANDARD

at some point, hopefully...

EXTEND OTHER COMPONENTS

```
<super-fancy-card>  
  <h3>Fridolin Frontend</h3>  
  <p>Frontend developer</p>  
</super-fancy-card>
```

EXTEND HTML ELEMENTS

```
<link rel="import"  
      href="components/fancy-button.html">
```

```
<button is="fancy-button">Save</button>
```

SHOULD I CARE?

YES, BECAUSE

- You have solved the "widget problem" in every project differently.
- There are a lot of libraries which solve it (jQuery UI, Angular JS directives, ...).
- Now comes the **HTML standard solution** for it.

YES, BECAUSE

A future where you pick your HTML components like a backend library (eg. Serializer, Guzzle, ...) and do a `bower install` and lets you use this HTML component **has** to excite you.

YES, BECAUSE

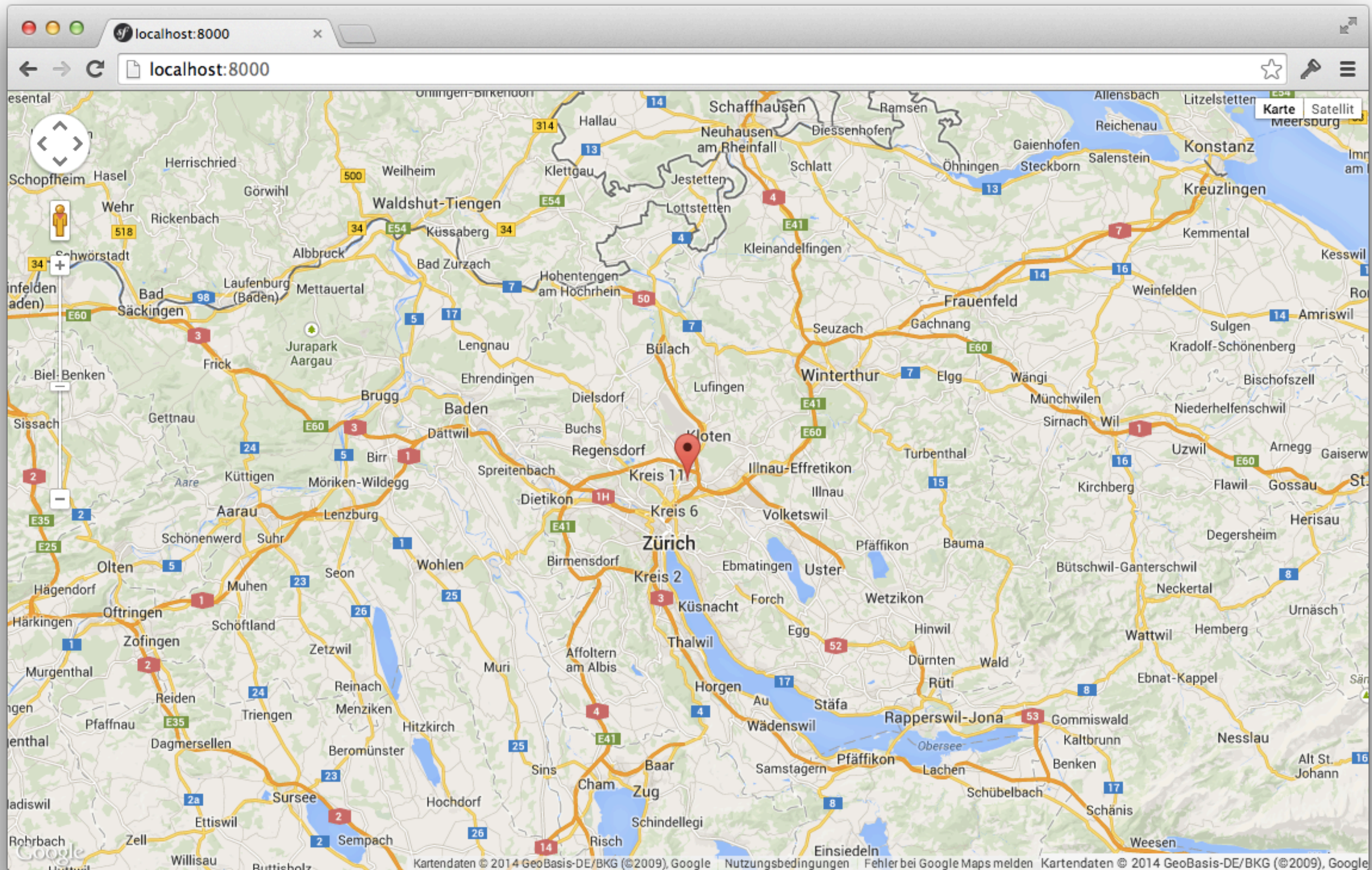
It will make your work easier, not harder. (Just a slight learning curve in the beginning, I promise ;)

YES, BECAUSE

You can start **now**. (Github uses it already in production :)

AS AN EXAMPLE: GOOGLE MAPS

```
<google-map latitude="37.779"  
  longitude="-122.3892">  
  <google-map-marker latitude="37.779"  
    longitude="-122.3892">  
    <h3>Hoi!</h3>  
  </google-map-marker>  
</google-map>
```

AS AN EXAMPLE: VOICE RECOGNITION

```
<voice-recognition id="recognition-element">  
</voice-recognition>
```

```
<script>  
document.querySelector('#recognition-element')  
  .addEventListener('result', function(e) {  
    console.log(e.detail.result);  
  });  
</script>
```

BROWSER SUPPORT AUGUST 2014

- **Chrome: Almost**
- **Firefox: Kind of**
- **Safari: Nope**
- **Internet Explorer: No way**

JQUERY FOR WEB COMPONENTS

Polyfill for web components, which allow web components even for Internet Explorer 9!

<https://github.com/Polymer/platform>

LIBRARIES

- Polymer from Google
 - Mozilla Bricks
 - X-Tags

POLYMER

- Based on platform.js like Mozilla Bricks.
 - On top of that: Binding
 - On top of that: Some sugar

LET'S START

SETUP

```
bower init
```

```
bower install --save Polymer/polymer
```

```
mkdir my-timeline
```

INDEX.HTML

```
<html>
<head>
  <script src="bower_components/platform/platform.js"></script>
  <link rel="import" href="my-timeline/my-timeline.html">

</head>
<body>
  <my-timeline>

</body>
</html>
```

MY-TIMELINE/MY-TIMELINE.HTML

```
<link rel="import" href="../../../bower_components/polymer/polymer.html">
```

```
<polymer-element name="my-timeline">
```

```
  <template>
```

```
    <div>
```

```
      <div id="segment"></div>
```

```
    </div>
```

```
  </template>
```

```
  <script>
```

```
    Polymer('my-timeline')
```

```
  </script>
```

```
</polymer-element>
```

TEMPLATES (OR SHADOW DOM)

The HTML in the template is not visible from outside. Also, the id does not collide with outside id.

CSS FOR THE ELEMENT

```
:host {  
  border: 1px solid grey;  
}  
#segment {  
  background-color: grey;  
  display: block;  
}
```

This CSS is not visible from outside. (Shadow DOM)

STYLE ELEMENTS FROM OUTSIDE

```
#my-timeline {  
    border: 1px solid blue;  
}
```

BUT... HOW DO I STYLE THE SEGMENT?

Shadow DOM is in general not stylable from outside. There are some non-standardized ways to "cross" into shadow DOM:

```
#my-timeline {  
    border: 1px solid blue;  
}  
#my-timeline::shadow #segment {  
    background-color: red;  
}
```

BETTER: REDESIGN ELEMENT

```
<style>
#my-timeline {
  border: 1px solid blue;
}
#my-timeline-segment {
  background-color: red;
}
</style>

<my-timeline>
  <my-timeline-segment>
</my-timeline>
```

ATTRIBUTES IN LIGHT HTML

```
<my-timeline duration="10">  
  <my-timeline-segment start="2" end="3">  
</my-timeline>
```

ATTRIBUTES IN JS

```
<polymer-element name="my-timeline" attributes="duration">
  <script>
    Polymer('my-timeline', {
      duration: 5,
      durationChange: function() {
        // duration has changed
      }
      domReady: function() {
        // upgraded HTML elements are ready
      }
    });
  </script>
</polymer-element>
})
```

POLYMER: BINDINGS

```
<my-backend-segment duration="{{duration}}" start="{{start}}" end="{{end}}">  
<my-timeline duration="{{duration}}">  
  <my-timeline-segment start="{{start}}" end="{{end}}">  
</my-timeline>
```

my-backend-segment encapsulates the backend API. Attributes are bound together.

DOESN'T THIS MAKE SENSE?

THX.