DEV PREVIEW

Understanding Polymer

The concepts and layering

Edit on GitHub

▶ **Table of contents**

Polymer isn't like other frameworks or libraries you might have used before. To fully understand what it is, you'll need to understand Polymer's world-view and take a quick tour through its three conceptual layers.

Custom elements: Polymer's world-view

Ever since the beginning of the web, browsers have shipped with a default set of elements. Most of them, like `<div>`, didn't do very much. But some elements are quite powerful. Consider the humble `<select>`. We take it for granted, but it's actually pretty impressive:

- **Functional**. The browser already knows what to do with a `<select>` element. When it encounters `<select>` in markup it creates an interactive control for the user.

- **Reusable**. The `<select>` element is a reusable package of functionality that you don't have to implement yourself.
- **Interoperable**. Every JavaScript library knows how to interact with DOM elements.
- **Encapsulated**. It keeps its internals all tucked away, so including one won't break the rest of your page.
- **Configurable**. You can configure its behavior with HTML attributes, without using any script.
- **Programmable**. If you grab the element from the DOM it also has methods and properties for things that don't make sense in markup.
- **Event Generator**. It dispatches events to let you know when something interesting happens.
- **Composable**. Not only can you include a `<select>` inside of most other kinds of element, its behavior can also change depending on which things you put inside of it.

Elements are pretty great. They're the building blocks of the web. Unfortunately, as web apps got more complex, we collectively outgrew the basic set of elements that ships in browsers. Our solution was to replace markup with gobs of script. In that shift, we've lost the elegance of the element.

Polymer returns to our roots. We think the answer is not gobs of script, but rather to build more powerful elements. A set of powerful new technologies called Web Components makes this possible.

That brings us back to the world-view of Polymer: Custom elements

When we say "element", we mean a real element, with all of the great properties of a built-in element. And why limit elements to UI? Some of the properties of elements are UI-specific, but most of them aren't. Elements can serve as a generic package for reusable functionality.

The world looks different when you take this view. You take a few low-level elements, put them together, and make a larger, more powerful element with its internals safely encapsulated. You can take those elements and build even bigger and better elements. Before you know it, you'll arrive at an entirely encapsulated, reusable *app*.

In the old world, script was your concrete, and the solution to most of your problems was to use gobs of it. In the new world, elements are your bricks; script is like mortar. Select the bricks that fit your needs most closely and use only a judicious amount of mortar to hold them together.

Layers of Polymer

There are three conceptual layers to Polymer:

1. **Web components**: Polymer is built on the Web Components standards. Not all browsers support these features yet, so the web components polyfill layer fills the gaps, implementing the APIs in JavaScript. At runtime, Polymer automatically picks the fastest path – native implementation or JavaScript.

2. **The Polymer library**: The Polymer library provides a declarative syntax that makes it simpler to define custom elements. And it adds features like two-way data binding, property observation, and gesture support to help you build powerful, reusable elements.

3. **Elements**: The Core and Paper element sets form a comprehensive set of UI and non-UI elements that you can use right out of the box. These elements depend on the Polymer library, but are separate and optional. You can use the elements without using Polymer directly, or you can use Polymer to create your own elements and not use the Core or Paper elements at all. You can mix and match the Core and Paper elements with other elements, including built-in elements and other custom elements.

Next steps

Now that you've got the basic concepts of Polymer, it's time to start digging in a bit more. Continue on to:

→ ABOUT CUSTOM ELEMENTS

Or jump straight to:

→ API DEVELOPER GUIDE

+POLYMER @POLYMER

/POLYMER FILE A BUG