POLYFILLS & MODERNIZR

What is a progressive enhancement?

Progressive enhancement is about creating web pages that…

1. Provide the **basic content and functionality** of web pages to everyone, regardless of their device or bandwidth.

2. Provide an **enhanced version** of web pages for those with more advanced devices or better bandwidth.

A simple CSS example of progressive enhancement would be a **box with round corners** (using CSS3 border-radius).

The quick brown fox jumped over the lazy dog. The quick brown fox jumped over the lazy dog.

All devices **should be able to access** the content in this box.

The quick brown fox jumped over the lazy dog. The quick brown fox jumped over the lazy dog.

Older browsers that do not support border-radius should see **a box with square corners**.

The quick brown fox jumped over the lazy dog. The quick brown fox jumped over the lazy dog.

Modern browsers that support border-radius, should see a **box with round corners**.

The quick brown fox jumped over the lazy dog. The quick brown fox jumped over the lazy dog.

What is a polyfill?

A polypill is a **piece of code** that provides the technology that developers expect the browser to provide natively.

Polypills are often used in situations where developers feel that progressive enhancement **would not be acceptable** for certain browsers or devices.

Polyfills can be used to **force browsers to support** all sorts of things including:

- HTML5 Elements
- HTML5 JavaScript APIs
- JavaScript Language Features
- CSS Selectors
- CSS Vendor Prefixes
- CSS Rendering

There are a **wide range** of polyfills available.

html5shim / html5shiv

http://code.google.com/p/html5shiv/

Selectivizr

http://selectivizr.com/

-prefix-free

http://leaverou.github.com/prefixfree/

Respond.js

https://github.com/scottjehl/Respond

Flexie

http://flexiejs.com/

CSS3 PIE

http://css3pie.com/

Should I use a polyfill?

CSS polyfills can **reduce development time**. They can allow developers to avoid hacks. They can even improve user experience.

But before using a polyfill in, you need to consider the **positives and negatives**.

For the particular problem I am trying to solve, is graceful degradation acceptable?

Yes: No need for a polyfill
No: Consider a polyfill

Would making my design work in all browsers require hacky markup, extra images, or dumbed-down code to modern browsers?

Yes: Consider a polyfill
No: No need for a polyfill

Would the polyfill create unacceptable performance or compatibility issues?

Yes: Avoid the polyfill
No: Consider a polyfill

What is Modernisr?

Modernizr is often mistaken for a polyfill. Unlike polyfills, it **does not fix problems**.

Modernizr is a collection of tests – or “detects” – which run as your web page loads. These detects allow you to use the results to **tailor the experience to the user**.

Modernizr detects a **wide range of features** including:

CSS animation
CSS gradients
CSS transforms
HTML5 file api
Desktop drag/drop
SVG
Web GL
Geolocation

Canvas
Web forms
Flexible box model
HTML5 history
Web sockets
Web workers
HTML5 input types
Application Cache

Modernizr reports this information by **injecting a specific series of class names** into the <html> element.

```
<html class="js no-flexbox no-canvas no-
canvastext no-webgl no-touch geolocation
postmessage no-websqldatabase no-indexeddb
no-hashchange no-history draganddrop
websockets rgba hsla multiplebgs
backgroundsize borderimage borderradius
boxshadow textshadow opacity cssanimations
csscolumns cssgradients cssreflections
csstransforms csstransforms3d csstransitions
fontface generatedcontent video audio
localstorage sessionstorage webworkers
applicationcache svg inlinesvg smil
svgclippaths placeholder">
```

This means adding the class for each feature when it is supported, and adding it with a **no- prefix** when it is not (e.g. .feature or .no-feature).

```
<html class="js no-flexbox no-canvas no-
canvastext no-webgl no-touch geolocation
postmessage no-websqldatabase no-indexeddb
no-hashchange no-history draganddrop
websockets rgba hsla multiplebgs
backgroundsize borderimage borderradius
boxshadow textshadow opacity cssanimations
csscolumns cssgradients cssreflections
csstransforms csstransforms3d csstransitions
fontface generatedcontent video audio
localstorage sessionstorage webworkers
applicationcache svg inlinesvg smil
svgclippaths placeholder">
```

For example, if you include Modernizr's detection for **CSS gradients**, the <html> element will either be given a class of “cssgradients” or “no-cssgradients”.

You can then write **specific CSS rules to cover both cases**. This could be a background image for older browsers and a gradient for modern browsers.

```
/* no gradient support */
.no-cssgradients .header
{
    background-image:
        url("image.png");
}

/* with gradient support */
.cssgradients .header
{
    background-image:
        linear-gradient(blue, green);
}
```

Should I use Mondernizr?

Modernizr makes it easy to **deliver tiered experiences**: make use of the latest and greatest features in browsers which support them, without leaving less fortunate users high and dry.

As Modernizr's main purpose is just to add class names to your <HTML> element, there are **relatively few downsides**, apart from the small download of the JS file and potentially one additional HTTP request.

Russ Weakley

Max Design

Site: maxdesign.com.au

Twitter: twitter.com/russmaxdesign

Slideshare: slideshare.net/maxdesign

Linkedin: linkedin.com/in/russweakley