Prologue
hand-written documents

data is markup

MOSAIC
X Window System • Microsoft Windows • Macintosh
data + template = document
**data** + template = document

document + JS is content
data + template is content
data is here
If you want more freely available linked data from the server markup, you should see this as an opportunity, not a threat.
Where We Are Today
Platform Capabilities

defined largely in terms of C++ implementation
Platform Capabilities

exposed via <markup>
and DOM bindings
<markup>

C++ DOM

JS Bindings

Render Tree

chasm
<form action="/people">
  <input type="text" name="first" placeholder="First Name">
  <input type="text" name="last" placeholder="Last Name">
  <button type="submit">Create</button>
</form>
4.10.22.4 Constructing the form data set

The algorithm to **construct the form data set** for a form `form` optionally in the context of a submitter `submitter` is as follows. If not specified otherwise, `submitter` is null.

1. Let `controls` be a list of all the **submittable elements** whose `form owner` is `form`, in **tree order**.
2. Let the `form data set` be a list of name-value-type tuples, initially empty.
3. **Loop**: For each element `field` in `controls`, in **tree order**, run the following substeps:
   1. If any of the following conditions are met, then skip these substeps for this element:
      - The `field` element has a `datalist` element ancestor.
      - The `field` element is `disabled`.
      - The `field` element is a `button` but it is not `submitter`.
      - The `field` element is an `input` element whose `type` attribute is in the **Checkbox** state and whose `checkedness` is false.
      - The `field` element is an `input` element whose `type` attribute is in the **Radio Button** state and whose `checkedness` is false.
      - The `field` element is not an `input` element whose `type` attribute is in the **Image Button** state, and either the `field` element does not have a `name` attribute specified, or its `name` attribute's value is the empty string.
      - The `field` element is an `object` element that is not using a **plugin**.
   
   Otherwise, process `field` as follows:
2. Let `type` be the value of the `type` IDL attribute of `field`.
3. If the `field` element is an `input` element whose `type` attribute is in the **Image Button** state, then run these further nested substeps:
   1. If the `field` element has a `name` attribute specified and its value is not the empty string, let `name` be that value followed by a single \".\" (U+002E) character. Otherwise, let `name` be the empty string.
   2. Let `name_x` be the string consisting of the concatenation of `name` and a single U+0078 LATIN SMALL LETTER X character (x).
   3. Let `name_y` be the string consisting of the concatenation of `name` and a single U+0079 LATIN SMALL LETTER Y character (y).
   4. The `field` element is `submitter`, and before this algorithm was invoked the user indicated a coordinate. Let `x` be the x-component of the coordinate selected by the user, and let `y` be the y-component of the coordinate selected by the user.
Let \( name_x \) be the string consisting of the concatenation of \( name \) and a single U+0078 LATIN SMALL LETTER X character (x).

Let \( name_y \) be the string consisting of the concatenation of \( name \) and a single U+0079 LATIN SMALL LETTER Y character (y).

The field element is submitter, and before this algorithm was invoked the user indicated a coordinate. Let \( x \) be the \( x \)-component of the coordinate selected by the user, and let \( y \) be the \( y \)-component of the coordinate selected by the user.

Append an entry to the form data set with the name \( name_x \), the value \( x \), and the type \( type \).

Append an entry to the form data set with the name \( name_y \) and the value \( y \), and the type \( type \).

Skip the remaining substeps for this element: if there are any more elements in controls, return to the top of the loop step, otherwise, jump to the end step below.

Let \( name \) be the value of the field element’s name attribute.

If the field element is a select element, then for each option element in the select element whose selectedness is true and that is not disabled, append an entry to the form data set with the name as the name, the value of the option element as the value, and type as the type.

Otherwise, if the field element is an input element whose type attribute is in the Checkbox state or the Radio Button state, then run these further nested substeps:

1. If the field element has a value attribute specified, then let value be the value of that attribute; otherwise, let value be the string "on".

2. Append an entry to the form data set with name as the name, value as the value, and type as the type.

Otherwise, if the field element is an input element whose type attribute is in the File Upload state, then for each file selected in the input element, append an entry to the form data set with the name as the name, the file (consisting of the name, the type, and the body) as the value, and type as the type. If there are no selected files, then append an entry to the form data set with the name as the name, the empty string as the value, and application/octet-stream as the type.

Otherwise, if the field element is an object element: try to obtain a form submission value from the plugin, and if that is successful, append an entry to the form data set with name as the name, the returned form submission value as the value, and the string "object" as the type.

Otherwise, append an entry to the form data set with name as the name, the value of the field element as the value, and type as the type.

If the element has a dir name attribute, and that attribute’s value is not the empty string, then run these substeps:

1. Let dirname be the value of the element’s dir name attribute.

2. Let dir be the string "ltr" if the directionality of the element is 'ltr', and "rtl" otherwise (i.e. when the directionality of the element is 'rtl').
2. Let \( name_x \) be the string consisting of the concatenation of \( name \) and a single U+0078 LATIN SMALL LETTER X character (x).
3. Let \( name_y \) be the string consisting of the concatenation of \( name \) and a single U+0079 LATIN SMALL LETTER Y character (y).
4. The field element is submitter, and before this algorithm was invoked the user indicated a coordinate. Let \( x \) be the x-component of the coordinate selected by the user, and let \( y \) be the y-component of the coordinate selected by the user.
5. Append an entry to the form data set with the name \( name_x \), the value \( x \), and the type type.
6. Append an entry to the form data set with the name \( name_y \) and the value \( y \), and the type type.
7. Skip the remaining substeps for this element: if there are any more elements in controls, return to the top of the loop step, otherwise, jump to the end step below.

4. Let \( name \) be the value of the field element's name attribute.
5. If the field element is a select element, then for each option element in the select element whose selectedness is true and that is not disabled, append an entry to the form data set with the name as the name, the value of the option element as the value, and type as the type.

6. Otherwise, if the field element is an input element whose type attribute is in the Checkbox state or the Radio Button state, then run these further nested substeps:
   1. If the field element has a value attribute specified, then let value be the value of that attribute; otherwise, let value be the string "on".
   2. Append an entry to the form data set with name as the name, value as the value, and type as the type.

7. Otherwise, if the field element is an input element whose type attribute is in the File Upload state, then for each file selected in the input element, append an entry to the form data set with the name as the name, the file (consisting of the name, the type, and the body) as the value, and type as the type. If there are no selected files, then append an entry to the form data set with the name as the name, the empty string as the value, and application/octet-stream as the type.

8. Otherwise, if the field element is an object element: try to obtain a form submission value from the plugin, and if that is successful, append an entry to the form data set with name as the name, the returned form submission value as the value, and the string "object" as the type.

9. Otherwise, append an entry to the form data set with name as the name, the value of the field element as the value, and type as the type.

10. If the element has a dir attribute, and that attribute's value is not the empty string, then run these substeps:
   1. Let dirname be the value of the element's dir attribute.
   2. Let dir be the string "ltr" if the directionality of the element is 'ltr', and "rtl" otherwise (i.e. when the directionality of the element is 'rtl').
Custom Control?
Button

Enhances standard form elements like buttons, inputs and anchors to themeable buttons with appropriate hover and active styles.

Examples

- Toggle
- Default functionality
- Checkboxes
- Icons
- Radios
- Split button
- Toolbar
### .serialize()

**Description:** Encode a set of form elements as a string for submission.

**Returns:** String

### .serialize()

This method does not accept any arguments.

The `.serialize()` method creates a text string in standard URL-encoded notation. It operates on a jQuery object representing a set of form elements. The form elements can be of several types:

```html
<form>
  <div><input type="text" name="a" value="1" id="a" /></div>
  <div><input type="text" name="b" value="2" id="b" /></div>
  <div><input type="hidden" name="c" value="3" id="c" /></div>
  <textarea name="d" rows="8" cols="40">4</textarea>
  <div><select name="e">
    <option value="5" selected="selected">5</option>
    <option value="6">6</option>
  </select></div>
</form>
```
The `serialize()` method can act on a jQuery object that has selected individual form elements, such as `<input>`, `<textarea>`, and `<select>`. However, it is typically easier to select the `<form>` tag itself for serialization:

```
$(form).submit(function() {
  alert($(this).serialize());
  return false;
});
```

This produces a standard-looking query string:

```
a=1&b=2&c=3&d=4&e=5
```

**Warning:** selecting both the form and its children will cause duplicates in the serialized string.

Note: Only "successful controls" are serialized to the string. No submit button value is serialized since the form was not submitted using a button. For a form element’s value to be included in the serialized string, the element must have a `name` attribute. Values from checkboxes and radio buttons (inputs of type "radio" or "checkbox") are included only if they are checked. Data from file select elements is not serialized.

**Example:**

Serialize a form to a query string, that could be sent to a server in an Ajax request.
Appeal to Magic
• People understand the core value propositions of the web
  ▶ Sharing
  ▶ Bookmarking
  ▶ Linking
• They work hard to emulate these characteristics
Discourse Demo
There's a general sense of dissatisfaction with the status quo.
Developers

"so close yet so far"
Users frustrated by the lack of perfect platform emulation in web apps
This gives people a sense that the web is not competitive.

If it made you happy, you're missing the bigger picture.
Doomsdayers

make platform stewards misunderstand
the nature of the malaise

It doesn't need to be an existential threat for us to take it seriously.
Turing Escape Hatch
or Turing Tarpit
add .active?
add `.active`?

what about desktop?
add `.active`?
what about desktop?
abandon `:active`?
reimplement again?
This should be described in terms of a primitive that user code can hook into.
The :active pseudo-class applies while an element is being activated by the user.

For example, between the times the user presses the mouse button and releases it. On systems with more than one mouse button, :active applies only to the primary or primary activation button (typically the "left" mouse button), and any aliases thereof.

There may be document language or implementation specific limits on which elements can become :active.

Selectors doesn't define if the parent of an element that is :active is also in that state.

Note: An element can be both :visited and :active (or :link and :active).

8.3. The input focus pseudo-class :focus

The :focus pseudo-class applies while an element has the focus (accepts keyboard or mouse events, or other forms of input).

There may be document language or implementation specific limits on which elements can acquire :focus.

8.4. The drag-and-drop pseudo-classes

The drag-and-drop pseudo-classes apply while the user is "dragging" or otherwise conceptually dragging for which the element is a valid drop target.

The :active-drop-target pseudo-class represents an element that is the current drop target for an element currently being dragged in a drag-and-drop interface.

The :valid-drop-target pseudo-class represents an element that is a possible drop target for an element currently being dragged in a drag-and-drop interface.

The :invalid-drop-target pseudo-class represents an element that is a possible drop target, but does not...
Destroys Visibility

In the aggregate, this kind of thing destroys visibility into what content is being presented to users.
What's Wrong?
This mental model leads to this kind of specification: defined in terms of the implementation, with JS tacked onto the side.
Fundamental Theorem of the Platform
Markup begets JavaScript objects via a parser
This mental model, which matches the platform developer's mental model, leads to things like :active being defined in terms of primitives that platform developers can hook into.
Better is being able to just replace what you need. To plug your own bits to a structure that supports you instead of making you re-create *everything* above any layer you want to shim something into.
Path for Natural Platform Evolution

If we define the platform's features in the same way as the imperative extensions, we provide a path for new declarative forms to develop and gain support naturally.
1. platform <markup>

2. imperative extensions

3. new "slang" <markup>

4. broad acceptance
If we define the platform's features in the same way as the imperative extensions, we provide a path for new declarative forms to develop and gain support naturally.
Case Study
App Cache
What is App Cache?
You need to log in before you can comment on or make changes to this bug.

Anne  2011-11-05 17:35:38 UTC

We should have a new section in the manifest that allows for files that should always be requested when online, but not when offline. The master entry should be allowed there. That way you can be sure dynamically generated pages are always up to date, while also still working offline.

Anne  2011-11-05 17:39:06 UTC

Comment 1
See also: http://www.w3.org/2011/web-apps-wg/papers/Facebook.html and https://gist.github.com/1341809

Anne  2011-11-05 18:43:08 UTC

Comment 2
Apparently Microsoft requested this before in 13160. To be clear, you cannot rely on the HTTP cache because:
1. It is more likely to go away.
2. Resources used by other pages will not be cached (e.g. help pages).
3. Using a manifest gives you control over complete application including its expiry time. You also do not have to perform conditional requests to see if any of the resources has been updated meanwhile.

Anne  2011-11-05 18:43:29 UTC

Comment 3
Sorry, meant to write bug-13160 to make it link.

Yehuda Katz  2011-11-05 21:34:25 UTC

Comment 4
Additionally, Microsoft’s semantics are not quite right. When the user agent is offline, you *do* want to use the master entry. This makes it a full-on offline app, with better updating semantics when the user agent is online for many use-cases.

In light of that, [NETWORK] is not exactly the right API.

Adrian Bateman [MSFT]  2011-11-05 22:58:41 UTC

Comment 5
We were trying to add the functionality while still being interoperable with current implementations. You could add the master page to the fallback section. We also have scenarios where you wouldn’t want that (for example, where you are trying to optimise connected applications and not support disconnected).

We're certainly open to suggestions for better alternatives.
In light of that, [NETWORK] is not exactly the right API.

Adrian Bateman [MSFT] 2011-11-05 22:58:41 UTC
We were trying to add the functionality while still being interoperable with current implementations. You could add the master page to the fallback section. We also have scenarios where you wouldn’t want that (for example, where you are trying to optimise connected applications and not support disconnected).

We're certainly open to suggestions for better alternatives.

Ian 'Hixie' Hickson 2011-11-11 00:29:45 UTC
So this is basically requesting an entry similar to FALLBACK except that the resource is only used if it can’t be fetched, even if it is already cached the network is still hit.

I don’t understand why this is better than the HTTP cache. If the idea is to hit the network when you can, but fall back to the cache when the resource is either not expired or the network is down, then that seems like exactly what local HTTP caching is for. Appcache is only useful because it lets you get to the data _before_ you check to see if it’s up to date.

Adrian Bateman [MSFT] 2011-11-11 16:43:08 UTC
This is better than the HTTP cache for all the reasons why AppCache needs to exist at all: to provide a predictable life time for the caching of content.

AppCache contains this functionality already with FALLBACK. The problem is that the master entry cannot currently participate in the FALLBACK contract. This means a news site that I want to enable while disconnected is always shown out of date even when connected.

Michael 2011-11-11 20:09:51 UTC
Solving this problem would go a long way to "fixing" the appcache and make broader adoption possible. Imho, this is the biggest piece of low hanging fruit to take care of and the most often requested change.

> So this is basically requesting...

The way i understand this request is to allow pages retrieved over the network to utilize the resources in an application cache for subresource loads, without itself being added to that application cache. The point is to speed up subresource loads.

The automatic pinning of 'master' is often not desired. And once added to the
Problem

No ability to use platform capability before demonstrating use case to spec editor
Scenario Solving
No Primitive
Navigation Controller
• What happens when the user loads a page?
• For the first time?
• Subsequent times?
• How does the cache work?
navigation

Controller ---- fulfill request for HTML ---- Window

Cache
Controller

Cache

Window

navigation

fulfill request for HTML

assets
Controller

Cache

Window

navigation

fulfill request for HTML
• The controller is a regular **Web Worker** using normal **MessageChannel**

• The controller operates online and **offline** after first request

• It can decide what **atomic resources** it needs to fetch before fulfilling

• It can fill the **cache** using a strategy of its choosing

• It can fetch cross-domain **Opaque Responses**
Declarative Form
An API to the Controller
{  "expiration": 300,
   "cache": ["index.html", "index.js", "index.css"]
}
{  
    "expiration": 300,
    "cache": [{
        "url": "index.html",
        "etag": "84ba9f"
    }, {
        "url": "index.js",
        "last-modified": "Wed, 1 May 2013 04:58:08 GMT"
    },
    "index.css"]
}
{
    "expiration": 300,
    "cache": ["index.html", "index.js", "index.css",
                "show_bug_handler.html"],
    "urlhandlers": [{
        "url": "show_bug?id=*",
        "page": "show_bug_handler.html"
    }]
}
Evolution

The default navigation controller will use the capabilities exposed to the navigation controller to implement the above features. Extensions to the manifest format could be implemented in libraries, gain broad adoption, and perhaps eventually be rolled into the platform.
Case Study

Web Components
• What is an Element?
• How does the parser create DOMElements?
• What would it mean to create a custom Element?
• What would it mean to subclass an Element?
• How are control Elements implemented?
Markup begets JavaScript objects via a parser
Proposal

Write a Recommendation that outlines a design philosophy for the web platform