

# Web Components as Micro Apps

---

**Craig West**

Talk resources and slides at:

**<https://wpjs.co.uk/ndc>**

The demo repo in this talk is a set of lessons on Vanilla JS Web Components and there is a YouTube video series explaining each lesson.

# Web Components as Micro Apps

---

Web Components as MicroApps

Or

HTML5

# Already built in to HTML5

---

- Re-render on prop change.
- Emit Custom Events and data.
- Listen to Custom Events.
- Expose methods.
- Cache API to preload/store pages.
- Store data with Local Storage (sync) and IndexedDB (async).

# In this talk...

---

- What is a Web Component?
- How do we use them?
- How do we make them:
  - - Vanilla JS
  - - with build tools (not frameworks).
  - - using frameworks to export app as a Web Component.
- How to import them into any framework.
- Using Web Components as interconnected Micro Apps/Micro Services.
- Use them as business widgets for not-tech users.

# Two Business Applications ( 1 )

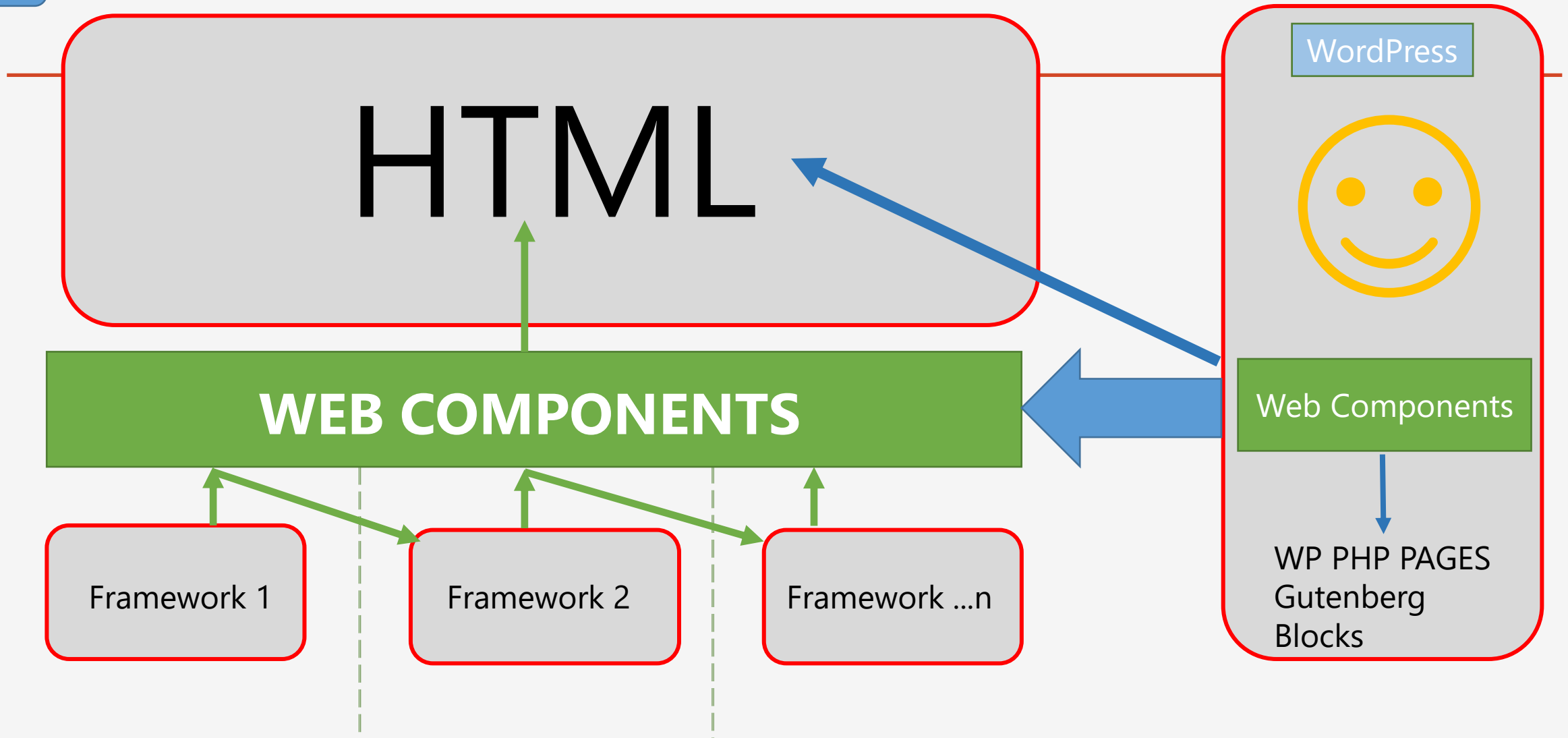
---

## Case One:

Disassemble a very monolithic PHP/MySQL into a set of HTML/JS/CSS Web Components that can be used in the app or as a Micro App in any HTML site.

This will be WordPress as it is very monolithic, PHP and very popular.

1



# Two Business Applications ( 2 )

---

## Case Two:

Entrepreneur wants to create a ONE-STOP tech conference site that uses several other businesses to provide information on tech events, booking, flights, hotels and tour guides.

Each Micro App is a separate concern, and all needed to be orchestrated together with minimal coding.

# International JS Conference London 2018 – Key Note Speech

<https://www.youtube.com/watch?v=1KJQurcLLdw>

Web Components & Micro Apps: Angular, React & Vue peacefully united?



International  
**JavaScript**  
Conference

**Angular, React, Vue  
and Co. - peacefully  
united thanks to Web  
Components and Micro  
Apps**

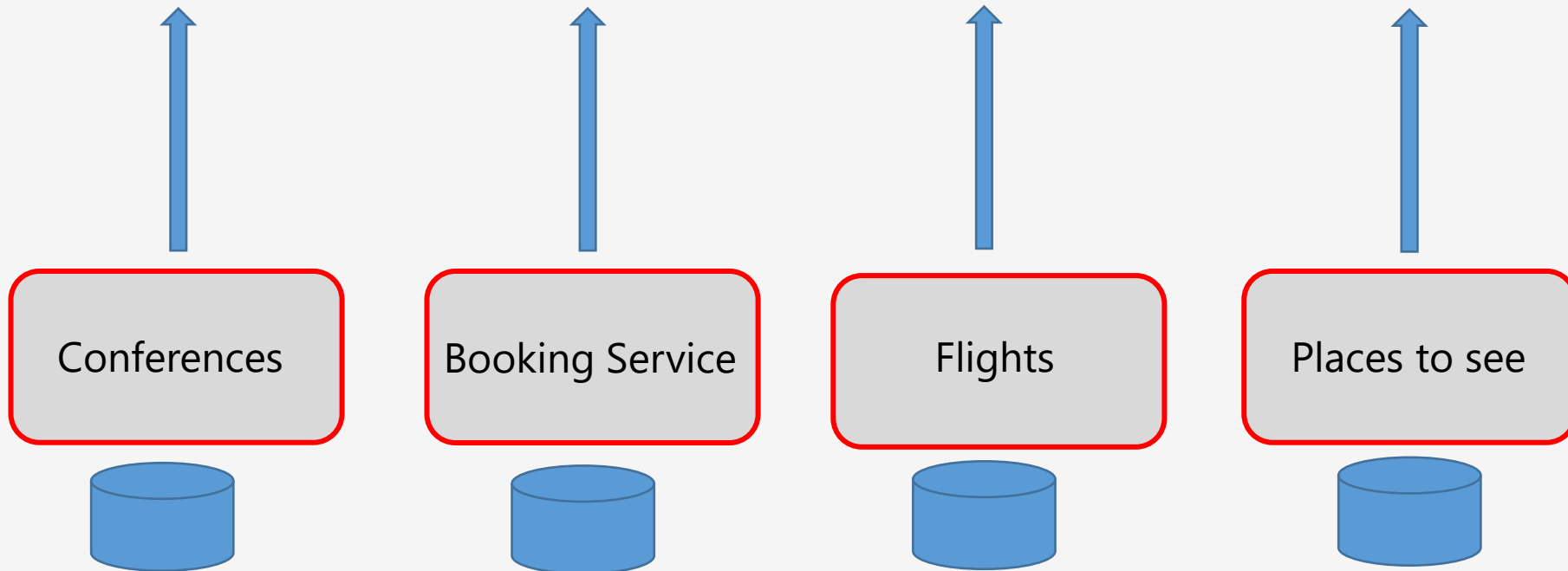
**Manfred Steyer**  
SOFTWAREarchitekt.at





2

A website that enables a user to find all tech conferences, book, get flight, hotel, travel guides etc...



**Micro Apps based on a variety of frameworks**

## Two Business Applications ( 2 )

---

`<div id="react-app"/>`

`<div id="vue-app" />`

`<div id ="angular-app" />`

`<div id="wordpress-app" />`

**Communicate events and data between all of them.**

## Two Business Applications ( 2 )

---

```
<!-- EVENT LOGGER COMPONENT -->
<iws-event-listener></iws-event-listener>
<!-- MICRO APPS -->
<iws-list-conferences></iws-list-conferences>
<iws-events-booking></iws-events-booking>
<iws-event-flights></iws-event-flights>
<iws-places></iws-places>
<!-- MICRO APPS -->
```

## Pseudo code

---

```
<list-conferences tech="JS" workshops="yes">
```

```
<book-an-event event="22" use="invoice" />
```

```
<show-flights city="LON" when="..."/>
```

```
<list-hotels min-stars="3" meals="yes" />
```

Declarative rather than Imperative

# Trainer of PWAs, Web Components & Async JS



Source:Wikimedia Commons



Source:Wikimedia Commons



1980: At University, I studied Chemistry and used MS-DOS, client/server main frame systems, ZX-Spectrum computer and Jupiter Ace recreationally.

Former careers as accountant SQL Server DBA and Business Information Architect.

And a plumber!



Source:Wikimedia Commons



Brighton, UK



Source:Wikimedia Commons

# Definitions

---

## **Micro App:**

A micro app is an interactive software module designed to perform like a full coded application or website. (Wikipedia)

## **Micro Service:**

A self-contained piece of business functionality with clear interfaces - "Do one thing and do it well". (Paraphrased from Wikipedia) or...

Which begs the question...

---

When is an app an app?

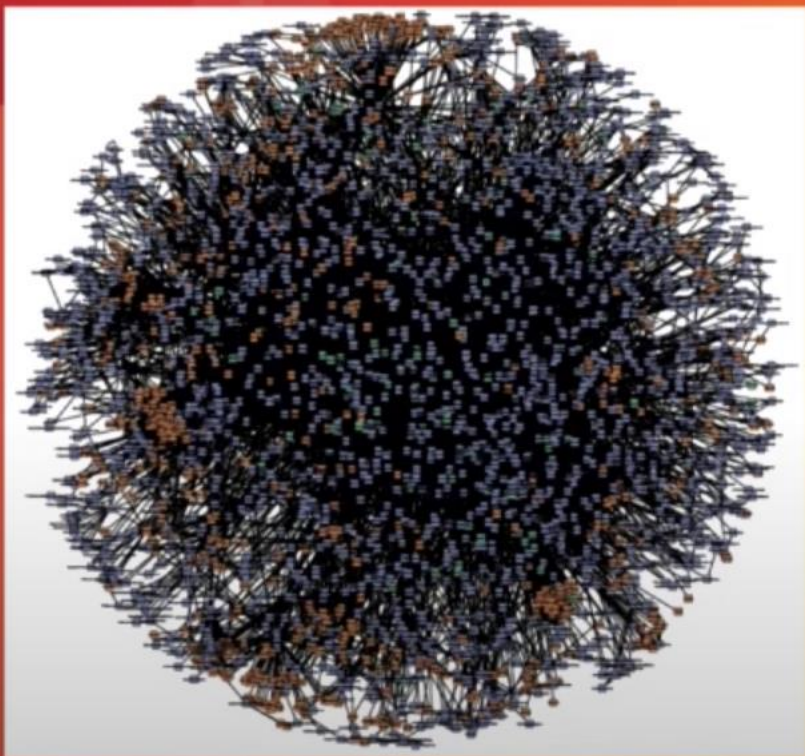
When is an app an micro app?

On own it is an app, with others a micro app?

Context...?



## Microservices at Amazon



Service-Oriented Architecture  
(SOA)

Single-purpose

Connect only through APIs

Connect over HTTPS

"Microservices"



**Darin Briskman**  
**AWS Developer Evangelist**

© 2018, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Single-Purpose, connecting through APIs over HTTPS = MICRO SERVICE



# What is HTML5?

---

<https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/HTML5>

HTML5 is the latest evolution of the standard that defines HTML. The term represents two different concepts. It is a new version of the language HTML, with new elements, attributes, and behaviors, **and** a larger set of technologies that allows the building of more diverse and powerful Web sites and applications. This set is sometimes called HTML5 & friends and often shortened to just HTML5.

- Create our own HTML tags.
- Re-render tags on state change.
- Preload/save web pages with the Cache API.
- Save data to Local Storage (synchronously).
- Store large data in an asynchronous, transactional database with IndexedDB.
- We can create a device level web and DB server.

# What is an **anchor** tag?

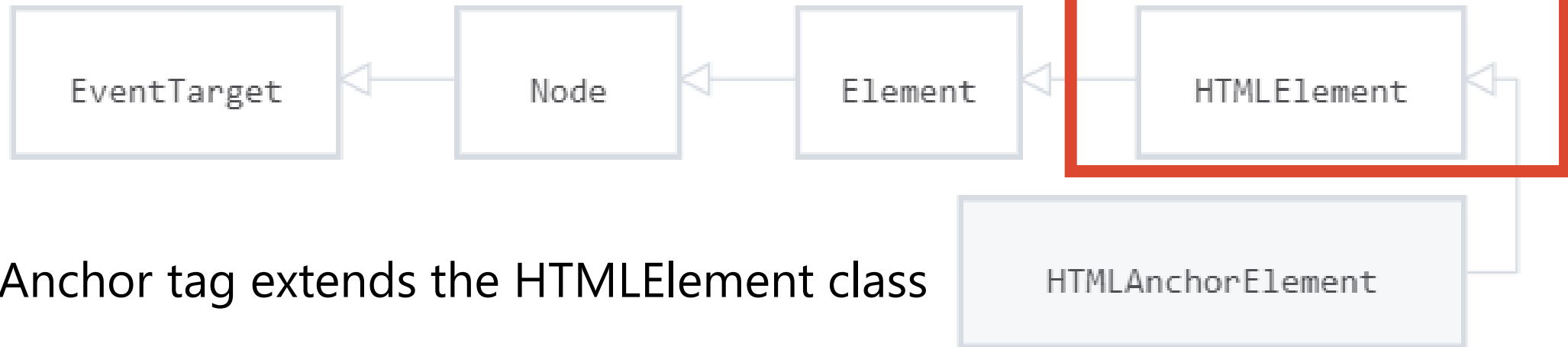
---

`<a href="#" target="_blank">CLICK EVENT</a>`

props

built-in click method

slot api



Anchor tag extends the HTMLElement class

<https://developer.mozilla.org/en-US/docs/Web/API/HTMLAnchorElement>

# What is an `input` element?

---

The **HTML `<input>` element** is used to create interactive controls for web-based forms in order to accept data from the user; a wide variety of types of input data and control widgets are available, depending on the device and user agent. The `<input>` element is one of the most powerful and complex in all of HTML due to the sheer number of combinations of input types and attributes.

<https://developer.mozilla.org/en-US/docs/Web/HTML/Element/input>

```
<input type="number" id="age" name="age" required  
value="32" min="0" max="10" step="1">
```

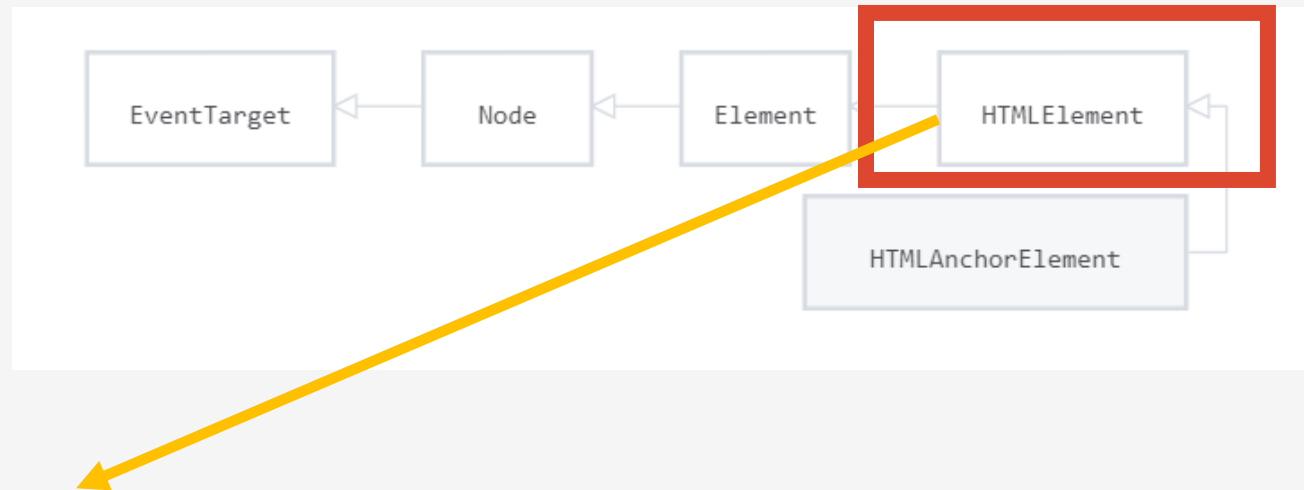
# What is a Custom HTML Tag?

---

```
<my-component> </my-component>  
<script src="file.js"> </script>
```

Must contain at least one hyphen.

Use just like any HTML tag.



They extend the HTMLElement class.

# How to use

---

For example, to define a mobile drawer panel, `<app-drawer>` :

```
class AppDrawer extends HTMLElement {...}  
window.customElements.define('app-drawer', AppDrawer);
```

To use the new tag:

We can create elements as part of HTML5 spec.

```
<app-drawer></app-drawer>
```



Using a custom element is no different to using a `<div>` or any other element. Instances can be declared on the page, created dynamically in JavaScript, event listeners can be attached, etc.

<https://www.webcomponents.org/introduction>

# How to use – Props Down

---

PROPS DOWN <show-post postid="34" />

Demo: 06-library/01-ndc-show-post/showPost.js

```
// List which attributes/props we wish to monitor as by default,  
// none are to save resources  
static get observedAttributes() {  
  return ['postId'];  
}  
  
// Do something if observed attribute has changed  
attributeChangedCallback(attributeName, oldValue, newValue) {  
  // this will fire initially as the element has no attribute  
  // but is added when page runs  
  if (attributeName === 'postId') { doSomething();}
```

# How to use – Events Up

---

## EVENTS UP

Demo: 05-events/15-child-to-parent

We use JS Custom Events to send event and data.

```
// ++++++ CUSTOM EVENTS ++++++
this.dispatchEvent(
  new CustomEvent('childOneClick',
    {detail: eventData}) ); // code edited for talk
// ++++++ CUSTOM EVENTS ++++++
```

In page, listen for the **'childOneClick'** event...

# How to use

---

NB Methods and properties on components can be directly accessed.

This would be a design choice as components tend to use props down events up, rather than sibling to sibling.



# How to use

---

```
<!-- OUTPUT OF CUSTOM ELEMENT -->  
<!-- We can add events to our Custom HTML tag just like regular HTML tags -->  
<my-component onclick="alert('Event Click')"></my-component>
```

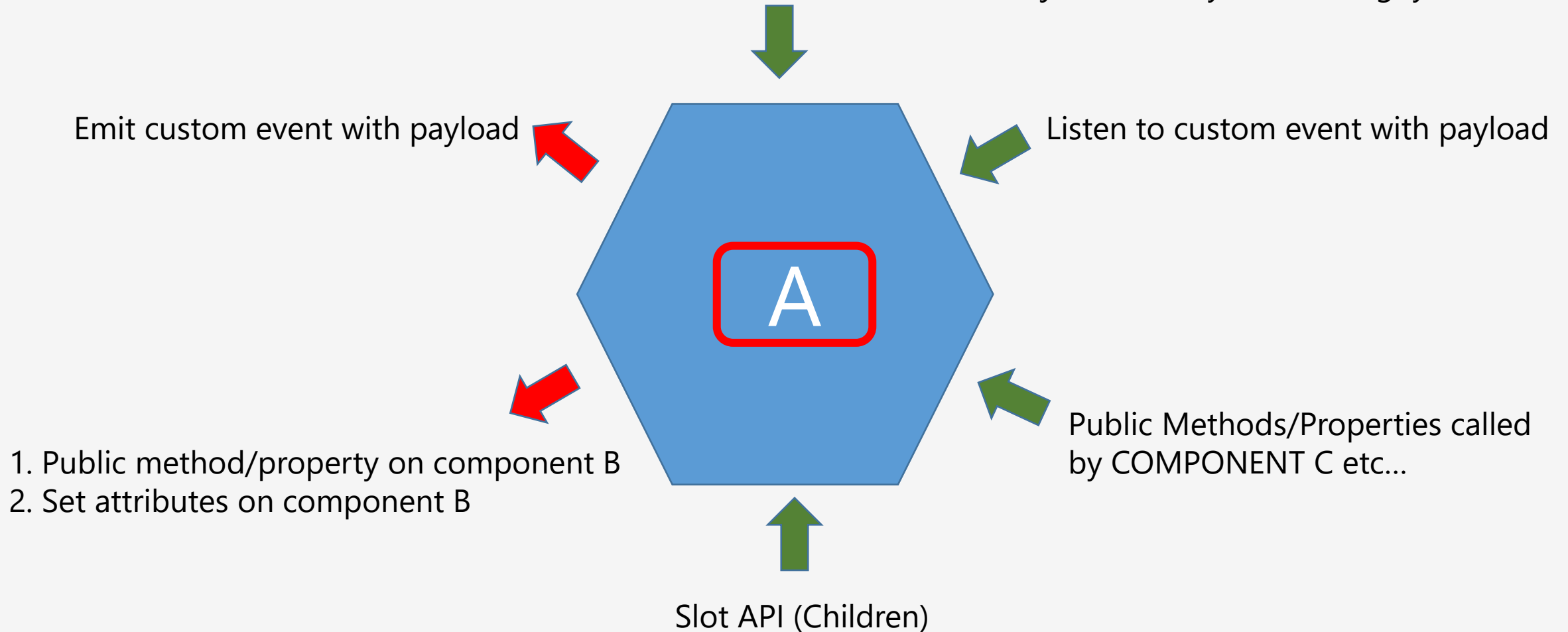
```
// we can now get a reference to the Custom HTML tag  
var customComponent = document.querySelector('my-component');  
btnMethod.addEventListener('click', function () {  
    customComponent.publicMethod(Math.floor(Math.random() * 1000));  
});
```

**Framework:** `<div id="app"> </div>`   `<web-component id="app" custom="{}" ...> </web-component>`






























---

HTML attributes (props)

Attribute can be string, number, array, object – usually JSON.stringify.



# Browser Support - Polyfills available

Browser support	 CHROME	 OPERA	 SAFARI	 FIREFOX	 EDGE
 HTML TEMPLATES	 STABLE	 STABLE	 STABLE	 STABLE	 STABLE
 CUSTOM ELEMENTS	 STABLE	 STABLE	 STABLE	 STABLE	 STABLE
 SHADOW DOM	 STABLE	 STABLE	 STABLE	 STABLE	 STABLE
 ES MODULES	 STABLE	 STABLE	 STABLE	 STABLE	 STABLE

# Encapsulation!

---

- Very important.
- Stumbling block in past.
- Now fully implemented with the Shadow DOM.
- We will see it in action in the demos but CSS is fully scoped with no bleeding between Light and Shadow DOM.
- Rules to allow crossing of boundary for CSS and JS...

# Shadow DOM – explained more fully later...

---

```
<div class="spacer"></div>
▼ <div id="output">
  <!-- OUTPUT OF CUSTOM ELEMENT -->
  ▼ <my-component>
```

FRAGMENT

→ ▼ #shadow-root (open)

```
▶ <style>...</style>
  <div>This is the SHADOW DOM</div>
```

```
▼ <slot>
```

↳ #text reveal

```
</slot>
```

"SLOT TEXT HERE"

```
</my-component>
```

**SHADOW DOM**



**SLOT projected into SHADOW**



# Who uses Web Components?

---

- Google 20,000+.
- Google Accelerated Mobile Pages.
- Ionic and Ionic/ReactJS/Angular (Ionic Apps with Capacitor can run on IOS, Android, Web and Electron). 'Write once, run anywhere..'

Salesforce has purchased Slack

- Salesforce – Lightning Web Components

Components can help you, like with interoperability, reuse and design systems. Learn how Salesforce uses Lightning Web Components to power their ecosystem of 5 million developers.

Google video about Web Components and Salesforce: [https://www.youtube.com/watch?v=YBwgkr\\_Sbx0](https://www.youtube.com/watch?v=YBwgkr_Sbx0)

# Types of Web Components

---

These are only limited by our use of JS as they are regular JS components.

We will look at couple of UI components and many more highly functional components that have built in functionality...

# Types of Web Components

---

- Fetch requests.
- Lazy loading and dynamic loading (scripts/components).
- Storing JSON in IndexedDB and rendering it via templates.
- Authentication and storage of JSON Web Tokens.
- Added offline capability and 'instant' pages.



## Useful references

---

<https://www.webcomponents.org/>

- main reference site

<https://custom-elements-everywhere.com/>

(Making sure frameworks and custom elements can be BFFs )

<https://webcomponents.dev/>

- over 40 libraries and compilers

*let's see these libraries and compilers...*

# Demo time

---

1 . Walkthrough of Web Components using repo.

2. Business case (1) site:

<https://wpjs.co.uk/demo1>

*Also in repo as 06/20-ndc/site*

<https://github.com/iwswordpress/ndc-london-web-components>

3. Business case (2) site:

<https://wpjs.co.uk/demo1>

# Converting Frameworks to Web Components

---

- Angular: <https://angular.io/guide/elements>
- Vue: <https://www.npmjs.com/package/vue-custom-element>
- Vue: <https://github.com/karol-f/vue-custom-element#demo>
- React: <https://github.com/LukasBombach/react-web-component>
- *Many others: <https://webcomponents.dev/>*

# Deployment

---

How to deploy Web Components as a third party:

- In HTML pages as we have seen.
- Via NPM we can import into builds.
- For Angular, Vue, React etc, there are well documented procedures, (e.g. <https://stenciljs.com/docs/framework-bindings>)

# Summary

---

Craig West  
wpjs.co.uk  
craig@wpjs.co.uk