ONGIS SCHOOL
FRONTEND DEV

SESI

1. KONSEP WEB COMPONENTS & TASK RUNNER
2. CUSTOM ELEMENTS
3. SHADOW DOM
4. HTML IMPORT
5. HTML TEMPLATE
6. FINISHING PROJECT

SESI 2 :
CUSTOM ELEMENTS

CUSTOM ELEMENTS

Mengijinkan Frontend Developer untuk menetapkan type baru untuk tag HTML

Contoh tag standar :

<h1></h1>, <header></header>, , dll

Contoh custom tag :

<el-head></el-head>, <aloveb-header></aloveb-header>

CUSTOM ELEMENTS

Custom Elements tidak otomatis ada pada HTML standar, perlu campur tangan Frontend Developer untuk mendaftarkan element baru, yaitu dengan beberapa cara:
Membuat element HTML / DOMmembuat element HTML / DOM baru berdasarkan tag yang sudah ada

membuat satu paket logic fungsi / perintah ke dalam satu tag

Menambahkan API dari DOM element bawaan

REGISTER NEW ELEMENT

Custom elements dibuat dengan document.registerElement();

```
var NewEl = document.registerElement('new-el')
document.body.appenchild(new NewEl());
```

argument pertama dari document.registerElement() adalah tag name yang kita buat namanya harus mengandung dash (-)
contoh :

```
<new-el></new-el>
```

tidak disarangkan

```
<fehead></fehead> atau <fe_head></fe_head>
```

REGISTER NEW ELEMENT

argument pertama adalah standar penulisan yang memungkinkan parser untuk membedakan element khusus dari element biasa, juga memastikan kompatibilitas ke depan saat tag baru ditambahkan ke HTML.

argument kedua adalah object (opsi) untuk menggambarkan prototype element. Ini adalah tempat untuk menambahkan fungsionalitas khusus (seperti public properties atau methods) ke tag html buatan.

REGISTER NEW ELEMENT

Secara default, custom element mewarisi dari dasar HTMLElement. Jadi contoh sebelumnya setara dengan :

```
var NewEl = document.registerElement(‘new-el’, {
    prototype: Object.create(HTMLElement.prototype)
});
```

Argumen ke document.registerElement (’NewEl') mengajarkan browser tentang elemen baru, dan mengembalikan konstruktor yang dapat Anda gunakan untuk membuat contoh <new-el>. Sebagai alternatif, Anda dapat menggunakan teknik instantiating elemen lain jika Anda tidak ingin menggunakan konstruktor.

EXTENDING ELEMENT

Custom elements memungkinkan untuk menambahkan fungsi element HTML pada tag asli yang ada dan element khusus lainnya. Untuk menambahkan fungsi, perlu menambahkan nama pada argument registerElement() dan protote element yang akan diwarisi

Contoh

Extending native elements

Kalau kita tidak puas dengan fungsi tag html

```
<button></button>
```

EXTENDING ELEMENT

kita boleh menambahkan sendiri fungsi pada tag html. Untuk menambahkan fungsi pada tag <button></button>, buat element baru yang mewarisi prototype HTMLButtonElement dan menambahkan fungsinya.

```
var FeButton =  document.registerElement('fe-button', {
                prototype: Object.create(HTMLButtonElement.prototype),
                extends: 'button'
});
```

panggil

```
<button is="fe-button"></button>
```

custom elements yang mewarisi dari elemen standar disebut extension custom elements, bisa kita baca dengan elemen button adalah element fe-button

LIFECYCLE CALLBACK METHODS

Elements dapat menentukan methods khusus untuk memanfaatkan masa keberadaan elements itu sendiri. Ada 4 jenis argument pada methods ini : masing masing punya nama dan tujuan tertentu

1. createdCallback
   contoh dari element yang dibuat

2. attachedCallback
   contoh dari element yang ditambahkan ke dalam document

3. detachedCallback
   contoh dari element yang dihilangkan dari dalam document

4. attributeChangeCallback(attrName, oldVal, newVal)
   attribute ketika menambahkan, menghapus, atau mengupdate

INTEGRASI PROJECT

STRUKTUR FOLDER PROJECT:

project

-------- dist

-------- src

---------- css

---------- sass

---------- style.sass

---------- components

---------- _file.sass

---------- js

---------- file.js

sass:

1. Tambahkan line

```
@import components/new-el.sass
```

2. buka folder components
3. buat file baru dengan nama _new-el.sass
4. Tambahkan line

```
new-el
        background-color:
        min –height: 100vh

        p
```

html:

1. Buka file index.html,
2. tambahkan line

```
<new-el></new-el>
```

Javascript internal:

1. Tambahkan line

   <script>

   </script>

   Di atas line </body> pada file index.html

2. Tambahkan line

```
var NewElement = Object.create(HTMElement.prototype);
NewElement.createCallback =  function() {
        this.innerHTML = ”Hei, kenalin saya new-el element baru buatan”;
};
var NewEl = document.registerElement(’new-el’, {prototype: NewElement});
```

Javascript external:

1. Create file new-el.js di folder js
2. Tambahkan line

```
var NewElement = Object.create(HTMElement.prototype);
NewElement.createCallback =  function() {
        this.innerHTML = ”Hei, kenalin saya new-el element baru buatan”;
};
var NewEl = document.registerElement(’new-el’, {prototype: NewElement});
```

NEXT ???
SHADOW DOM

CREDIT:

http://webcomponents.org

http://nodejs.org

http://npmjs.com

http://jshint.com

http://sass-lang.com

GLOSARIUM :

vendor prefix : sebutan untuk penambahan beberapa karakter khusus di awal penulisan property html / css, tujuan auto-prefixer adalah sebagai standar uji coba lintas browser untuk property css baru