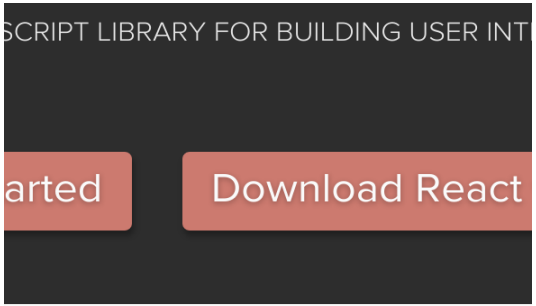
Фреймуърк системи за уеб програмиране

Лекция 4:

Уеб Компоненти

- HTML
- CSS
- JS



VIRTUAL DOM

React abstracts away the DOM from you, giving a simpler programming model and better performance. React can also render on the server using Node, and it can power native apps using React Native.

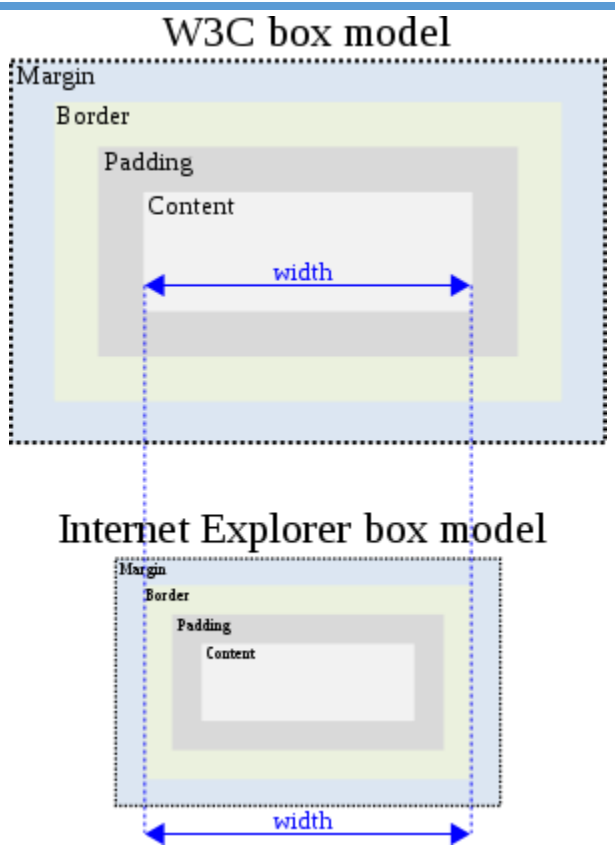
- ~~HTML~~
- CSS
- JS

GWT

- ~~HTML~~
- ~~CSS~~
- ~~JS~~

Има(ше) време, когато те бяха „задължителни“

The old Internet Explorer box model



https://en.wikipedia.org/wiki/Internet_Explorer_box_model_bug

The old Internet Explorer event model

Name	Description	Argument type	Argument name
attachEvent	Similar to W3C's addEventListener method.	String	sEvent
		Pointer	fpNotify
detachEvent	Similar to W3C's removeEventListener method.	String	sEvent
		Pointer	fpNotify
fireEvent	Similar to W3C's dispatchEvent method.	String	sEvent
		Event	oEventObject

https://en.wikipedia.org/wiki/DOM_events#Microsoft-specific_model

Netscape layer tag (what's Netscape?)

```
<LAYER ID=layer1 TOP=20pt LEFT=5pt BGCOLOR="#CC00EE"
WIDTH=200>
  <H1>Layer 1</H1>
  <P>Lots of content for this layer.</P>
  <IMG SRC=violets.jpg align=right>
  <P>Content for layer 1.</P>
  <P>More Content for layer 1.</P>
</LAYER>
```

http://web.archive.org/web/20040929103224/http://developer.netscape.com/docs/manuals/communicator/dynhtml/index.htm

Нямаше консистентен начин за интеракция с браузъра

Ситуацията сега е доста по-добра

SVG (basic support) - REC

Method of displaying basic Vector Graphics features using the embed or object elements. Refers to the SVG 1.1 spec.

Global	92.97% + 2.46% =	95.4
U.S.A.	93.7% + 1.19% =	94.8

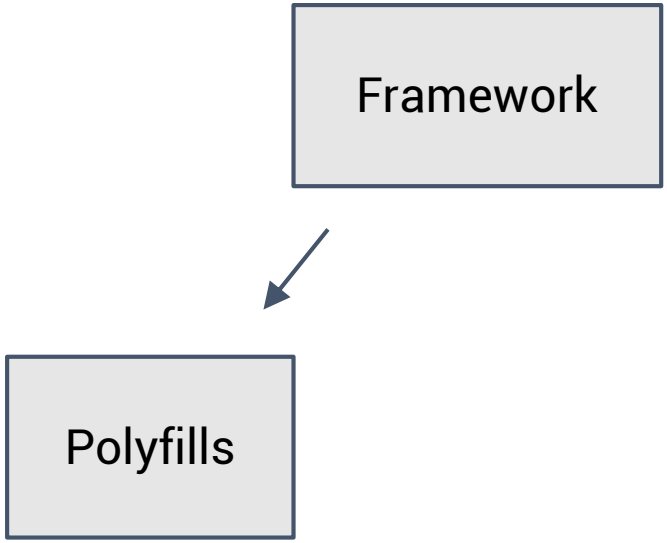
Current aligned | Usage relative | Show all

IE	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Chrome for Android
		31					1 4.1	
		36					1 4.3	
		37					4.4	
8		39					4.4.4	
2 9	31	40	7		7.1			
2 10	38	42	7.1					
2 11	39	43	8	30	8.3	8	40	42
2 Edge	40	44	9	31	9			
	41	45		32				
	42	46						

http://caniuse.com/#feat=svg

Polyfills

"import __future__" за web.



Запълва липсващи функционалности

А другите неща, които frameworks ни дават?

Абстракциите имат Течове

Original Java Stack Trace from GWT?

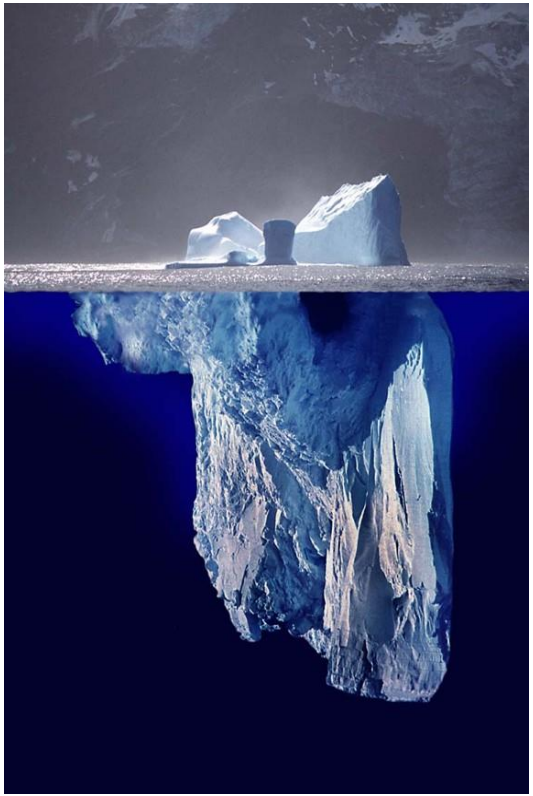
3

Is it possible for GWT to give the stack trace for the original Java code after a crash, as opposed to the JS stack trace?

I'm using Chrome 17.

java gwt stack-trace

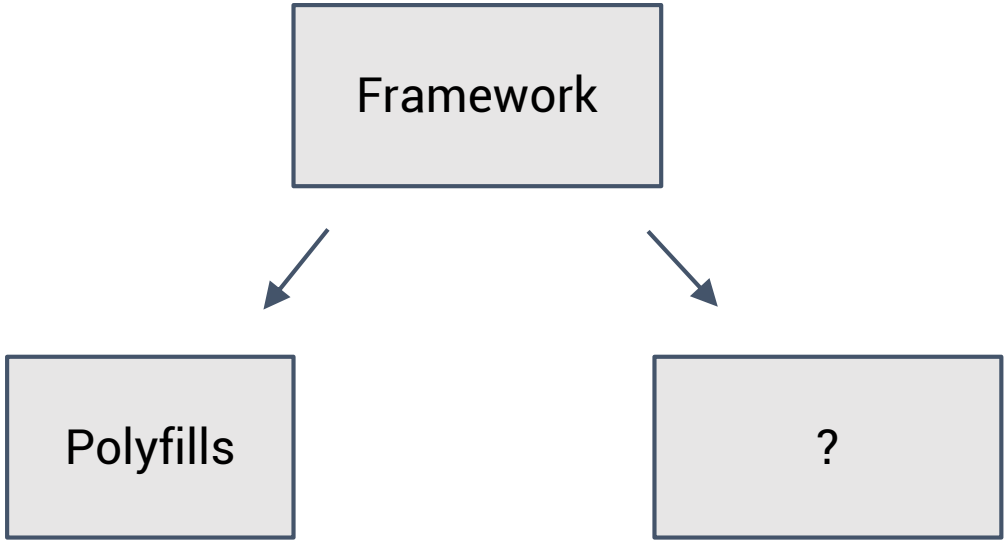
Абстракциите са Абстрактни



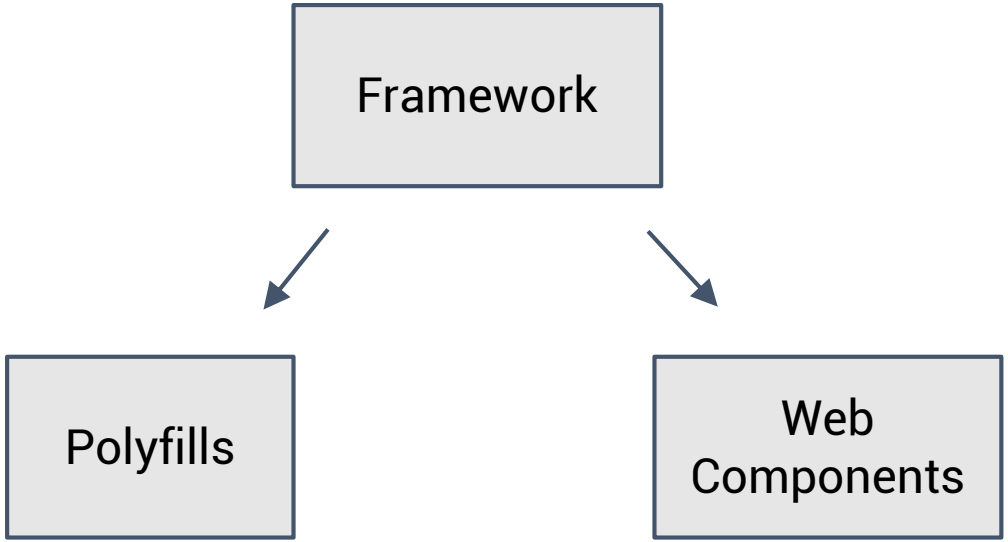
https://upload.wikimedia.org/wikipedia/commons/a/ac/Iceberg.jpg

Преизползване на кода?

Angular 1.0 -> 2.0



Организация на кода



Организация на кода

Web Components

- HTML Templates
- HTML Imports
- Custom Elements
- Shadow DOM

Web Components

- HTML Templates
- HTML Imports
- Custom Elements
- ~~Shadow DOM~~

HTML Templates

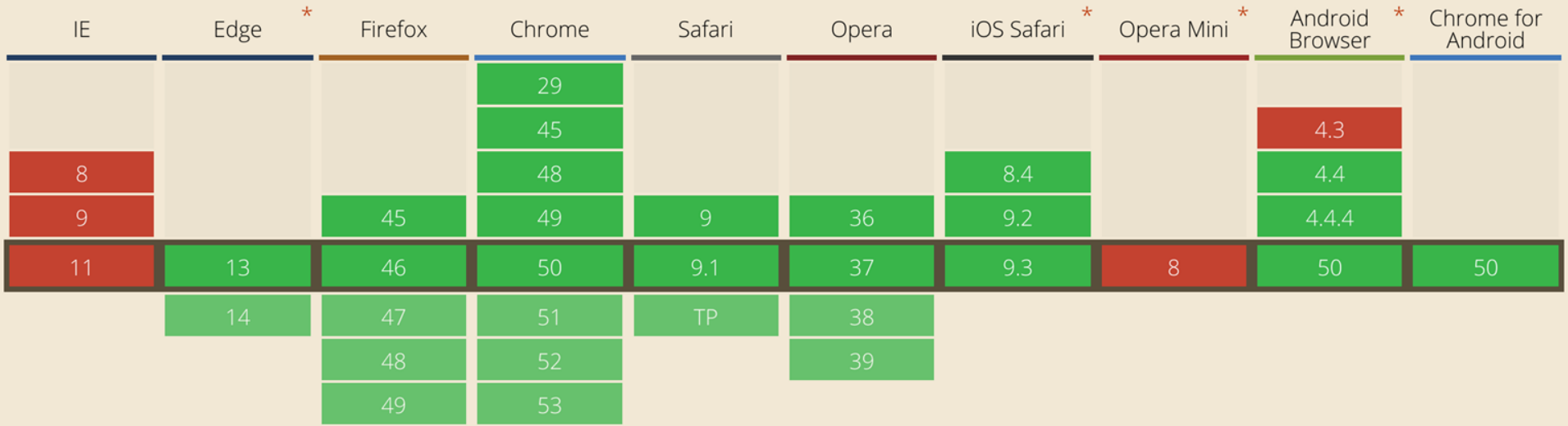
- Вече са част от HTML спецификацията

HTML templates - LS

Global 70.33%

Method of declaring a portion of reusable markup that is parsed but not rendered until cloned.

Current aligned | Usage relative | Show all



HTML Templates

- <template>...</template>
- Съдържанието се парси от браузъра.
- Съдържанието не се рендира (невидимо за потребителя).
- Съдържанието е скрито (невидимо за querySelector).
- Съдържанието е инертно. (JS не се изпълнява)

```
<template id=t>
  <div>Hello World!</div>
</template>

<script>
  var node = document.importNode(
      document.querySelector('#t').content, true);
  document.body.appendChild(node);
</script>
```

```
<template id=t>
  <div>Hello World!</div>
</template>

<script>
  var node = document.importNode(
      document.querySelector('#t').content, true);
  document.body.appendChild(node);
</script>
```

```
<template id=t>
  <div>Hello World!</div>
</template>

<script>
  var node = document.importNode(
      document.querySelector('#t').content, true);
  document.body.appendChild(node);
</script>
```

```
<template id=t>
  <div>Hello World!</div>
</template>

<script>
  var node = document.importNode(
      document.querySelector('#t').content, true);
  document.body.appendChild(node);
</script>
```

```
<template id=t>
  <div>Hello World!</div>
</template>

<script>
  var node = document.importNode(
      document.querySelector('#t').content, true);
  document.body.appendChild(node);
</script>
```

bit.ly/zf-template

HTML Imports

- <link rel="import" href="">
- Oddly slightly contentious.
- Brilliant for code organization, keep HTML, CSS, and JS together in one file.

HTML Imports

- Are de-dup'd
- Can contain anything an HTML file can contain (HTML, CSS, JS)
- Scripts are executed in the context of the window that imported them

import.html

```
<template id=t>
  <div>
    Hello World!
  </div>
</template>

<script>
  (function() {
    var doc = document.currentScript.ownerDocument;
    var node = document.importNode(
        doc.querySelector('#t').content, true);
    document.body.appendChild(node);
  })();
</script>
```

import.html

```
<template id=t>
  <div>
    Hello World!
  </div>
</template>

<script>
  (function() {
    var doc = document.currentScript.ownerDocument;
    var node = document.importNode(
        doc.querySelector('#t').content, true);
    document.body.appendChild(node);
  })();
</script>
```

import.html

```
<template id=t>
  <div>
    Hello World!
  </div>
</template>

<script>
  (function() {
    var doc = document.currentScript.ownerDocument;
    var node = document.importNode(
        doc.querySelector('#t').content, true);
    document.body.appendChild(node);
  })();
</script>
```

import.html

```
<template id=t>
  <div>
    Hello World!
  </div>
</template>

<script>
  (function() {
    var doc = document.currentScript.ownerDocument;
    var node = document.importNode(
        doc.querySelector('#t').content, true);
    document.body.appendChild(node);
  })();
</script>
```

import.html

```
<template id=t>
  <div>
    Hello World!
  </div>
</template>

<script>
  (function() {
    var doc = document.currentScript.ownerDocument;
    var node = document.importNode(
        doc.querySelector('#t').content, true);
    document.body.appendChild(node);
  })();
</script>
```

index.html

```
<html>
  <head>
    <script src="webcomponents.js"></script>
    <link rel="import" href="import.html">
  </head>
  <body>
  </body>
</html>
```

bit.ly/zf-html-imports-tmpl

index.html

http://webcomponents.org

```
<html>
  <head>
    <script src="webcomponents.js"></script>
    <link rel="import" href="import.html">
  </head>
  <body>
  </body>
</html>
```

bit.ly/zf-html-imports-tmpl

index.html

```
<html>
  <head>
    <script src="webcomponents-lite.js"></script>
    <link rel="import" href="import.html">
  </head>
  <body>
  </body>
</html>
```

bit.ly/zf-html-imports-tmpl

index.html

```
<html>
  <head>
    <script src="webcomponents.js"></script>
    <link rel="import" href="import.html">
  </head>
  <body>
  </body>
</html>
```

bit.ly/zf-html-imports-tmpl

util.js

```
Context = function() {
  this.doc = (document.currentScript||document._currentScript).ownerDocument;
};

Context.prototype.import = function(id) {
  return document.importNode(this.doc.querySelector('#'+id).content, true);
};
```

import.html

```
<template id=t>
  <div class=greeting>
    Hello World!
  </div>
</template>

<script>
  (function() {
    var ctx = new Context();
    document.body.appendChild(ctx.import('t'));
  })();
</script>
```

bit.ly/zf-import-context

Custom Елементи

- Създаване на нови елементи, не наследяване на съществуващи.

```
var proto = Object.create(HTMLElement.prototype);

proto.createdCallback = function() {
  this.textContent = 'Hello World!';
};

proto.addMoreText = function(s) {
  this.textContent = this.textContent + s;
};

document.registerElement('hello-element', {
  prototype: proto
});
```

```
var proto = Object.create(HTMLElement.prototype);

proto.createdCallback = function() {
  this.textContent = 'Hello World!';
};

proto.addMoreText = function(s) {
  this.textContent = this.textContent + s;
};

document.registerElement('hello-element', {
  prototype: proto
});
```

```
var proto = Object.create(HTMLElement.prototype);

proto.createdCallback = function() {
  this.textContent = 'Hello World!';
};

proto.addMoreText = function(s) {
  this.textContent = this.textContent + s;
};

document.registerElement('hello-element', {
  prototype: proto
});
```

Custom Елементи Lifecycle Callbacks

createdCallback

attachedCallback

detachedCallback

attributeChangedCallback

```
var proto = Object.create(HTMLElement.prototype);

proto.createdCallback = function() {
  this.textContent = 'Hello World!';
};

proto.addMoreText = function(s) {
  this.textContent = this.textContent + s;
};

document.registerElement('hello-element', {
  prototype: proto
});
```

```
var proto = Object.create(HTMLElement.prototype);

proto.createdCallback = function() {
  this.textContent = 'Hello World!';
};

proto.addMoreText = function(s) {
  this.textContent = this.textContent + s;
};

document.registerElement('hello-element', {
  prototype: proto
});
```

```
<html>
<body>
        <hello-element></hello-element>
</body>
</html>
```

bit.ly/zf-custom-element

util.js

```
Context = function() {
  this.doc = (document.currentScript||document._currentScript).ownerDocument;
};

Context.prototype.import = function(id) {
  return document.importNode(this.doc.querySelector('#'+id).content, true);
};


function newElement(name, proto) {
  var ep = Object.create(HTMLElement.prototype);
  Object.keys(proto).forEach(function(key) {
    ep[key] = proto[key];
  });
  document.registerElement(name, {prototype: ep});
}
```

util.js

```
Context = function() {
  this.doc = (document.currentScript||document._currentScript).ownerDocument;
};

Context.prototype.import = function(id) {
  return document.importNode(this.doc.querySelector('#'+id).content, true);
};


function newElement(name, proto) {
  var ep = Object.create(HTMLElement.prototype);
  Object.keys(proto).forEach(function(key) {
    ep[key] = proto[key];
  });
  document.registerElement(name, {prototype: ep});
}
```

```
newElement('hello-element', {

  createdCallback: function() {
    this.textContent = 'Hello World!';
  },

  addMoreText: function(s) {
    this.textContent = this.textContent + s;
  },
});
```

bit.ly/zf-newelement