Capgemini

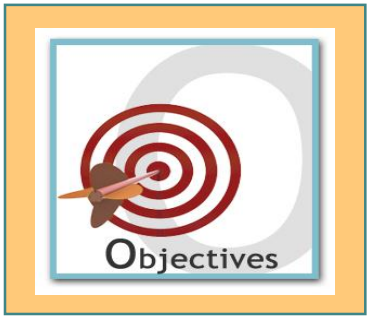
HTML5 & Its New Features

Lesson 6: The Offline Access & Drag and Drop API

Lesson Objectives

In this lesson you will learn about:

- Introduction to Offline Access in HTML5
- Techniques of implementing Offline Access
- Introduction to Drag & Drop API in HTML5
- Techniques of implementing Drag & Drop API
- Limitations of HTML5
- Comparison – Native SDK & HTML5



What is an offline Access?

- Web applications have become a major part of many people's lives
- It's becoming increasingly important for web-based applications to be accessible offline
- All browsers have caching mechanisms, but they're unreliable and don't always work as you might expect
- Wouldn't it be great if we could use them even when offline
- Until recently, there wasn't any viable way to do this
- However, HTML5 introduces new methods for enabling a web site or web application to function without a network connection

Need for an Offline Access

- Web applications are becoming more complex and capable every day
- There are many examples of web applications doing the same job as desktop applications in various fields
- For example Google Docs, Picasa, etc.)
- However, one major disadvantage is that they cannot work when the user is not connected to the Internet
- This is where HTML5's new offline storage comes in
- It tries to remove that disadvantage
- HTML5 achieves this by defining a way to store files in a cache, so that when the user is offline, the browser still has access to the necessary files
- These can be HTML, CSS or JavaScript files, or any other assets the site needs to run

How to implement an Offline Access in HTML5?

- At its simplest, an offline web application is a list of URLs
- This URL includes resources like HTML, CSS, JavaScript, images, or any other kind of resource
- The home page of the offline web application points to this list, called a Manifest file, which is just a text file located elsewhere on the web server
- A web browser that implements HTML5 offline applications will read the list of URLs from the manifest file
- It also downloads the resources, cache them locally, and automatically keep the local copies up to date as they change
- When the time comes that you try to access the web application without a network connection, your web browser will automatically switch over to the local copies instead

Offline Access – Current Browser Support

Android 2.0+

Chrome 5.0+

Firefox 3.5+

Opera 10.6+

iPhone 2.1+

Safari 4.0+

The Cache Manifest

- An offline web application revolves around a cache manifest file
- A manifest file is a list of all of the resources that your web application might need to access while it's disconnected from the network
- In order to bootstrap the process of downloading and caching these resources, you need to point to the manifest file, using a manifest attribute on your <html> element
- Your cache manifest file can be located anywhere on your web server, but it must be served with the content type text/cache-manifest
- Every one of your HTML pages points to your cache manifest file, and your cache manifest file is being served with the proper Content-Type header
- A manifest file can have any file extension, but needs to be served with the correct mime-type i.e. text/cache-manifest

Cache Manifest in html Tag

- **Referencing a manifest file**
  - **To enable the application cache for an app, include the manifest attribute on the document's html tag:**

```
<!DOCTYPE HTML>
<html manifest="/cache.manifest">
<body> ... </body>
</html>
```

- **The manifest attribute can point to an absolute URL or relative path, but an absolute URL must be under the same origin as the web application**

```
<html
manifest="http://www.example.com/example.manifest"> ...
</html>
```

Sections in Cache Manifest

- **CACHE**
  - **The CACHE section is considered the default - i.e., if no section heading has been defined, the browser will assume this is the CACHE section**
  - **Beneath this heading, you can list URIs to resources you want the browser to download and cache for offline use, including URIs hosted externally**

```
CACHE:
/html5/src/logic.js
/html5/src/style.css
/html5/src/background.png
http://example.com/css/widget.css
```

Sections in Cache Manifest

- **FALLBACK**
  - **The FALLBACK section tells the browser what to serve when the user tries to access an uncached resource while offline**
  - **It contains two values per line, separated by a space**
  - **The first value is the request URI to match, and the second is the resource sent upon matching**
  - **In below given example we're telling the browser that when an offline user requests a URI matching "/status.html", it should instead serve the cached file "offline.html"**

```
CACHE MANIFEST

FALLBACK:
/status.html /offline.html
/Two.html  /One.html
/Form.html /Message.html
```

Sections in Cache Manifest

- **NETWORK**
  - **Finally, we have the NETWORK section, used to tell the browser explicitly which resources are only available while online**
  - **By default, this uses the asterisk * symbol, meaning all resources that are not cached will require a connection**
  - **Alternatively we can whitelist specific url prefixes**

```
CACHE MANIFEST

NETWORK:
*
```

The Cache Manifest – Complete Example

```
CACHE MANIFEST

# This is a comment

CACHE:
/css/screen.css
/css/offline.css
/js/screen.js
/img/logo.png
http://example.com/css/styles.css

FALLBACK:
/ /offline.html

NETWORK:
*
```

Triggering Cache Refresh

- Once a cache has been successfully downloaded, the browser will retain it until either the user clears the cache or you trigger an update
- Triggering an update with your manifest file requires that the contents of that file change, not just the assets themselves
- Updating the assets on your server will not trigger a cache update
- You must modify the manifest file
- If you're adding or removing resources completely, you'll have to edit your manifest file anyway
- But what if you're just amending an already cached stylesheet?
- Just add in a simple version number comment that you change when you want to trigger an update

Triggering Cache Refresh

- **Example**

```
CACHE MANIFEST

# Version 9

CACHE:
/css/screen.css
```

- **The next time you want to trigger a cache refresh, just increment the version number**

- **When the user next visits the online version of a page including this manifest, it will re-download the manifest file, notice the change, download the listed assets, and purge the existing cache**

DEMO

- **Demonstration on implementing Offline Access in HTML5**

HTML5 – Drag & Drop API

- **We've been using libraries like JQuery and Dojo to simplify complex UI elements**
- **For example animations, rounded corners, and drag and drop**
- **Drag and drop is a first class citizen in HTML5!**
- **Drag and drop is one of the often used functionalities throughout websites for various reasons**
- **The spec defines an event-based mechanism, JavaScript API, and additional markup for declaring that just about any type of element be draggable on a page**
- **Browser support**
  - The latest versions of Firefox and Chrome support this API

How to use Drag & Drop API in HTML5?

- **Drag & Drop requires only a few things to work:**
  - **Something to drag**
  - **A drop target**
  - **JavaScript event handlers on the target to tell the browser it can drop**

- **The following elements are draggable by default:**
  - **<img> elements**
  - **<a> elements (with an href).**

- **If you want to drag a different element, you need to set the draggable attribute to true**

```
<div id="word1" draggable="true">the</div>
```

How to use Drag & Drop API in HTML5?

- **Starting a drag operation**
- **When a drag operation on a draggable object begins, a number of things need to be done:**
  - Define the drag effect that is allowed
  - There are three effects that can be allowed:
    - copy —indicates that the item being dragged will be copied
    - move —indicates that the item being dragged is to be moved
    - Link —indicates that some sort of relationship is to be created between the original item and the resultant one
- **Whilst the item is being dragged, the drag effects can be altered to indicate that certain effects are allowed at certain locations**
- **Set the data that is to be dragged using setData on the dataTransfer object**
- **Define the drag image using setDragImage, if it's to be customized**

DEMO

- **Demonstration on using Drag & Drop API in HTML5**

Limitations of HTML5

- The HTML5 spec is DRAFT and is in ongoing development (change)
- Video support is not standardized
- Not currently a single codec that all browsers will support
- Currently no support for cue points or alpha (transparent) video
- Limited desktop browser support
- Only the newest and best have reasonable support
- Internet Explorer will not have decent support until IE9
- Challenges getting consistent page display across browsers
- Graceful page degradation is potentially complex
- Currently no designer tools for creating HTML5 animation or interactivity (all must be implemented by a developer)
- Limited developer debugging tools

Comparison of Native SDK & HTML5

Factors	Native SDK	HTML5 - Cross Platform Framework
Development Language	Platform specific language needs to be used, like iPhone-objective c, Android-jave etc.	Common language, HTML/JSP, JS, CSS, needs to be used.
Deployment	Platform specific executable will be generated.	A common link will be used across all platforms.
Platform Support	Respective platform supported	Multiplatform support (with few limitations)
Native Feature Support	Access to all native features is possible	Limited access to native features
Development efforts	More. Different code for different platform	Less. One code base is sufficient for multiple platforms
Distribution	Application needs to be distributed platform wise	No need to distribute. Can be accessed using application link only
Cross-platform Support	Compile per target	Support multiple platform
Application performance	Excellent	Average, depending upon network availability
Developer Community & Support	Good	Growing

Summary

In this module, you have learnt:

- HTML5 introduces new methods for enabling a web site or web application to function without a network connection
- All browsers have caching mechanisms, but they're unreliable and don't
- Always work as you might expect
- An offline web application revolves around a cache manifest file
- It contains a list of resources to be stored for use when there is no network connectivity
- HTML5 comes with a Drag and Drop (DnD)
- API that brings native DnD support to the browser making it much easier to code up