Vodafone Position Paper for Offline WebApps Workshop

Vidhya Gholkar, Daniel Appelquist; Vodafone Group R&D 12 October 2011

Introduction

What is an offline Web application? We think it's a Web application that makes use of technologies available in the Web platform to be able to function and provide a robust user experience under varying network conditions, including the condition of having no Internet connectivity.

Why is this important in a world where we are increasingly online wherever we go? Because there are always places where you are offline or in zones of diminished network connectivity. This is physics.

Don't all WebApps need to be connected anyway? No, there are plenty of WebApps that can function perfectly well without a network. Even social apps can such as the mobile version of Twitter on the Web, can display the last known list of messages and allow the user to compose a message which is then saved for later sending when connection is reestablished.

A host of technologies have been developed and deployed to help solve this problem and provide a better development and deployment environment for these types of applications. They all have problems and they are all trying to solve slightly different problems. We believe there is value to coming to consensus on what we, as the Web community, what the offline Web app to look like, how we want it to behave and how we want to to be developed and deployed.

Toward Offline WebApps

In this position paper, our aim is to articulate our view on some of the use cases and technical issues that need to be clarified so that offline Web applications can be available across devices and OSs in the way Web sites are available today.

"Native Apps" are a particular class of offline applications that have proven to be commercially attractive as the store distribution model is a convenient way for the developer to monetise the application whilst providing an easy way for the user to find (discover) an application. Indeed it can be argued that these commercial imperatives have led to an explosion in both "cool" and innovation. **An app** is an independent piece of software which the user "owns" (arguably, a rather useful emotional/psychological need) and which the user expects to **work to some degree even when there is no connectivity**. **A website on the other hand is just that - a "site" far away; and the default expectation is that without connectivity there is no availability of any kind.** In addition, the default expectation for the Web is that when connectivity is available, the Web application is available **on any device where Internet connectivity is available.**

The next step, we think is that **Web apps should be available on any device even when the Internet is not available.**

For this to happen, an application, accessed via a URL, needs a manifest file to tell it what to have available locally to work offline. It also needs to know when connectivity is available again. At the simplest level, therefore, the format of the manifest file and status indicators need to be standard. Furthermore, the role of the manifest file when there is Web connectivity needs to be agreed. Some of this functionality already has been developed as part of HTML5. However, use of these technologies case cannot meet the “own” and “monetise” needs? Is a different mechanism needed to meet these needs?

Furthermore, when a Webapp is cached offline as with HTML5 Application Cache, in current implementations, the user has no easy way to know which applications are available when there’s no connectivity and which are not. The default assumption is likely to be that Web apps are not available.

What User Experience Do We Want for Off-Line WebApps?

Do we want to live in a world where the user makes an explicit choice to “install” a WebApp - leading to the emotional connection of the app being “on” the phone and also making it clear that the app can be launched when the user is not online? If so what else does or should installation mean?

- What degree of trust needs to be coupled with Installation? Is trust more than the right to use device capabilities?
- When and how are such rights granted? Prior to installation? After installation - first time or every time the application is “run”?
- Does the user spend money on downloading the “app” to find that it wants access to capability (s)he is not willing to give?

If a “container” mechanism similar to the W3C widgets or the Chrome .crx extension packaging is needed, then some standard container is needed to ensure that the Web package can be used on any Web accessible device. Are there additional needs for the packaging itself e.g. signing or is this an implementation issue?

In the end different types of WebApps will have different needs with regards to offline operation. We think multiple packaging formats, multiple manifest formats, multiple code-signing mechanisms, and multiple security and permission systems on the Web will lead to additional fragmentation and developer confusion and even a backlash against the Web. If there is to be One Web, then this Web should also extend itself offline through standardized formats, interfaces and and through user experiences that are consistent across implementations, devices and platforms.