Google Summer of Code 2016 - Mozilla

Project : Download app assets at runtime

Personal Details

Name :	Krishna Kannan
Email :	k.krish@yahoo.com, krishna.kay@gmail.com.
Telephone:	+65 83039960
Other Contact Method :	krish@comp.nus.edu.sg
Course of Study :	Master of Computing
University :	National University of Singapore - School of Computing
Country of Residence:	Singapore
Timezone:	+8 GMT
Primary Language:	English, Tamil

Brief

Firefox for Android is one of the most widely used browsers. The team is currently involved in reducing the size of the Android Package - APK. As a first step of this ongoing effort the fonts are excluded from being packaged along with the APK. The scope of reducing the size of the APK does not stop there. There are various opportunities and scopes for further reducing the size of APK such as excluding hyphenation dictionaries, translation and so on from being packaged. The task is to download the excluded assets from the server. This initiative can effectively reduce the size of the APK File.

Project Proposal

I propose to implement features that enables the browser to download the assets like hyphenation dictionaries at the runtime. The hyphen (-) is a punctuation mark used to join words and to separate syllables of a single word [1]. Sometimes it is required to break the words into parts instead of moving the entire word to the next line. Thus the words are broken down based on the syllables. The words are divided into the nearest breaking points [2]. A hyphenation algorithm is a set of rules (especially one codified for implementation in a computer program) that decides at which points a word can be broken over two lines with a hyphen [3].

Scope

Firefox Gecko supports the hyphenation while rendering the text in the browser. Support files are being used while rendering the hyphens and hyphenating the long words in a paragraphs. Firefox's rendering engine uses "Hyphenation Dictionaries". Currently, Firefox for Android is

packaging the Hyphenation Dictionaries in the APK while shipping the product. The principle goal of this project is to remove the Hyphenation Dictionaries from the APK as they eat up more space. The Hyphenation Dictionaries are to be downloaded at the runtime instead of being packaged along with APK. Along with this, I also propose to implement a helper class which can be used to smartly schedule the downloads based on the conditions and smartly reschedule the failed downloads. This helper class, once implemented can be used by other projects too.
I would like to approach the implementations systematically. Once the above tasks are completed, we can start researching about the other assets like localization files and how to remove them from the APK and download it at the runtime.

Preparatory Tasks & Observations

As initial steps I have downloaded the code and built the code on the local machine. While building the projects I have learnt more about the mach building tool and the building mode. I have also built the code using artifact-mode and normal local building process. I have also imported the project into the Android Studio and gone through the code and understood the basic flow of the code.

From the documentations I understood that this task is similar to the task that enabled the fonts to be downloaded at runtime. Therefore I have read through the bugs related to the above mentioned tasks. I have also started solving bugs related to the Android project.

After setting up the code, I was able to build the fennec. I observed that during the initial run of the browser the fonts are getting downloaded. The download content are prepared and downloaded. I also see setting checksum of the files being downloaded as they can be used to check whether the content is being tampered with. We can use the same technique to download the hyphenation dictionaries and check the integrity of the downloaded content. Without checking the integrity of the downloaded content we will be exposing the browser to threats.

I moved on to create a build without “Hyphenation Dictionaries” with the following command.

To remove from the local build - **export MOZ_EXCLUDE_HYPHENATION_DICTIONARIES=1** in mozconfig file.

To remove from the all the builds we can add export **MOZ_EXCLUDE_HYPHENATION_DICTIONARIES=1** in confvars.sh

First thing I noticed after including the flag is the Hyphenation folder was removed from **ObjDir/dist/bin/hyphenation**

With the hyphenation dictionaries successfully removed from the build, the code was imported and the Fennec was packaged to produce the APK file. Now the Fennec was running without the Hyphenation Dictionaries. An extensive testing was performed to make sure the browser

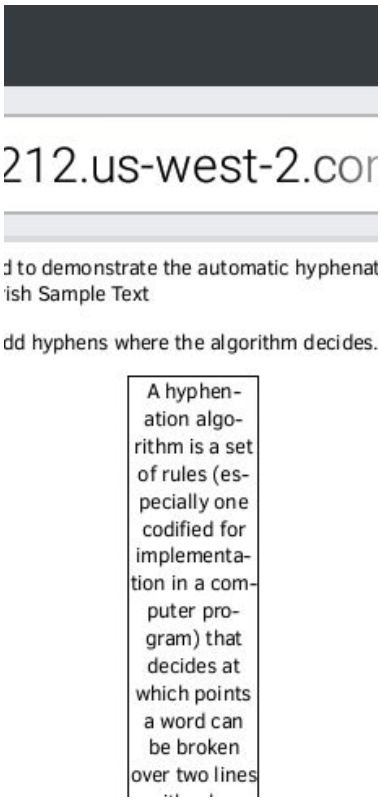
was not crashing. In this extensive testing there was no crashing. The pages were readable but the **hyphen support was not present**.

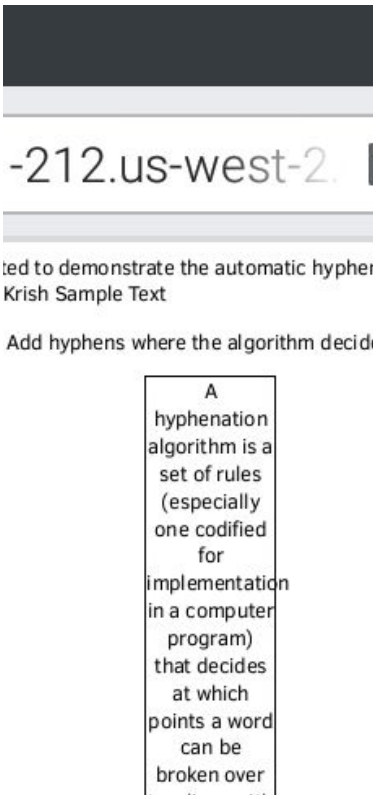
To test the behaviour of the browser, I have created a sample web application and hosted the application on AWS.

Hyphen-Test Sample Application Link

http://ec2-54-213-91-212.us-west-2.compute.amazonaws.com:8080/HyphenTest

Since the hyphenation dictionary supports many languages we can extend this sample application to support and display many languages. Currently in this demo for demonstration purposes I have created text consisting of Hyphens in Spanish and English





The above are the screenshots with and without the hyphenation dictionaries.

I have also learnt about the **Reftests.** I ran the default reftests to understand it better. Once after downloading the hyphenation dictionary, we can perform the reftests to verify the working of the downloaded assets. I have also created sample reftests which succeeded when hyphenation dictionary was present.

Created a sample reftest.list containing along with sample html files
== hyphen.html hyphen-ref.html

HTML Files
Hyphen html contains style tag containing
style="-webkit-hyphens: auto; -ms-hyphens: auto;hyphens: auto;" for <p> tag containing texts - "An extremely long word" and determined width.
Hyphen-ref html contains
"An ex-
tremely long word" and width applied to the <p> tag which produced the visually similar pages.

Success Case :
REFTEST TEST-PASS | http://192.168.200.251:8888/tests/custom-test/hyphen.html | image comparison (==)

Failure Case :
REFTEST TEST-UNEXPECTED-FAIL | http://192.168.200.251:8888/tests/custom-test/hyphen.html | image comparison (==), max difference: 255, number of differing pixels: 8220

I have also performed some random reftests to understand how it works and grasped the working of the Reftests.

Schedule of deliverables

April 15 - May 22 : Fixing related bugs, reading documentations and related/similar bug details.

May 23 - May 31 : Fixing the minor bugs related to the projects like packing hyphenation dictionaries for the reftests.

June 1 - June 30 : Completing the main tasks in Hyphenation Dictionary exclusion such as Modifying the DownloadContentServices and related files to support downloading .dic files, Adding dictionary to online catalog, and let Gecko to rebuild the list.

July 1 - July 7 : Thoroughly testing the build for any missing gaps, build failures, Application crashes.

July 8 - July 12 : Researching and experimenting with GcmNetworkManager, Alarm Manager, and Job Scheduler. Identification of compatibility of APIs and Compat versions of APIs.

July 13 - July 15 : Implementation of Helper class for scheduling and executing the background tasks with the approach decided during the research phase.

July 16 - July 20 : Extensive testing for the designed Scheduler.

July 21 - Aug 10 : Identification of bugs and researching the unknowns of the Project Localization files download at runtime.

Aug 10 - Aug 20 : Preparing a comprehensive report of the unknowns/bug fixes for the localization files download at runtime project.

Aug 20 - Aug 24 : Pencils down stage. Submitting the code written during the summer for evaluation.

Beyond : Once after completion of submitting the project for evaluation, I would like to continue to work on the project Localization for any extended bugs/ unknowns. I would also like to be associated with Mozilla and continue code for it after GSoC.

Open Source Development Experience :

I am an open source developer at Syslog-ng Project. syslog-ng is an open source implementation of the syslog protocol for Unix and Unix-like systems. My association with Syslog-ng started with the **Google Summer of Code - 2015**. I have developed an
Android Application - Syslog-ng Monitor for Mobile

https://pzolee.blogs.balabit.com/2015/09/monitor-syslog-ng-with-android-app

Application - https://github.com/Krishna41/Syslog-ng-Monitor-Android

The application serves the administrators to monitor the current status of the Syslog-ng Logging servers. The administrator can view status, configuration and other details of the server. The application also had a feature of saving many instance details and client certificate management and secure transport between the client and server.

Developed extensive jUnit test cases for the Android application and I have also developed a User-Interface Testing Project with **Robotium Testing Framework.**

Test Project - https://github.com/Krishna41/Syslog-ng-Monitor-Android-Test

I have also resolved some bugs in **Firefox for Android,** and resolving many bugs as an ongoing process in order to get a better insight of the code.

Work/Internship Experience:

After my under-graduation, I worked for **Verizon Inc**. as a software engineer where I developed a product which was basically a Tablet application, developed using C# .net. Verizon as one of the largest telecommunications company had an army of technicians to support the customers. To improve the work efficiency of the customers they developed a tablet application which will be carried by the technicians from door to door.
I then joined a startup in India which worked on solving problems in recommendation engines in marketplace. There I was responsible for the development of a social network application using java and Neo4j.

Currently I am pursuing my masters and interning at a research institute - Keio-NUS CUTE Center - http://cutecenter.nus.edu.sg/ as an **Android developer.** I am responsible for the application - CrowdTrails and National Heritage Trails supported by NHB Singapore.

CrowdTrails -

https://play.google.com/store/apps/details?id=sg.edu.nus.cutecenter.crowdtrails&hl=en

National Heritage Trails

https://play.google.com/store/apps/details?id=singapore.heritage.trails&hl=en

Academic Experience:

As a part of my Masters Program I have done many projects among which the following are my favorites.

Event Recommendation Systems - In this project Semi-Lazy Mining paradigm was used to get recommendations. Performed kNN Search along with Random forest classifier to improve the efficiency and accuracy of the recommendations. The dataset used for the project was obtained from many Event Based Social Networks.

Hire for a long haul - Analytics Project - Solutions that would help the employer identify potentially “best fit” candidate not just for the skillset but also for the longer association with the company.

Android Reverse Engineering - Bytecode Manipulation - ASM is a library used to generate, convert/modify and analyse the compiled Java Classes. Disassembled the APKs to obtain the dex and ultimately obtained the class files. Calculated the offsets and injected code into the class files and reassembled the code to produce the APKs and signed it with JarSigner.

Other than this I have also worked on a project to identify the community in a social graph and interaction between the communities / overlapping communities and determination of strength between the communities. Worked on a Network security project at undergraduate level to classify the malicious packets and identify the real source of the attack packets over a period of time.

Why Me :

I am a Graduate student at National University of Singapore, specializing in computer science. I did my undergraduate coursework in the field of Computer Science and Engineering at Anna University, India.

I have experience in Java, C# and Python. I have developed various android applications as a hobby and published it in Google Play store. Some of my hobby projects can be found here :

https://play.google.com/store/apps/developer?id=Spaghetti%20Stack&hl=en

During my undergraduate studies, I participated in **The Great Mind Challenge**, a nationwide software project contest conducted by IBM Academic Initiative. Our project was selected as one of the Top 20 Projects in the country where approximately more than 150,000 students participated.

I am very much involved and interested in open source development. I am a strong believer of Open source. I am a **RedHat Certified Engineer** and have been a *nix OS user for long time.
I believe I could be a match for the Firefox for android project, and it thrills me to have a thought of working on a project that can have real impact.

Why Mozilla :

The codebase of Mozilla is an Ocean, which is a perfect feast for a person like me, who is very much passionate and interested in coding and who believes in the cause of open source. I would happily be involved in a project that could reach several million people at free of cost. Mozilla created a dent in Web universe which can be felt by every web developers, and also always setting the bar high. Combined with this cause and my insatiable thirst for knowledge in mobile development I believe working for Mozilla on this project can be the best thing ever.

References

[1] https://en.wikipedia.org/wiki/Hyphen
[2] https://en.wikipedia.org/wiki/Hyphen#Separating
[3] https://en.wikipedia.org/wiki/Hyphenation_algorithm