**Readme**

**Synopsis:** The aim of this project was to understand and implement a Dynamo-style key-value storage; this assignment is about implementing a simplified version of Amazon’s Dynamo.

There are three main pieces you need to implement:

1) Partitioning,

2) Replication, and

3) Failure handling.

**Implementation:** I used an Android activity and Content Provider to implement the Dynamo Style key value storage. This was tested on five different Android emulators and was checked under scenarios like node failure, node joining and node network quorum check on partitioning.

**References:** The algorithm is based on the paper  
 <http://www.allthingsdistributed.com/files/amazon-dynamo-sosp2007.pdf>

**Running the application:**

1. Install the SimpleDynamo.apk and run it from each emulator instance.
2. Your app should have an activity used for testing.
   1. It has have five buttons, three buttons that displays “Put1”, “Put2”, and “Put3”, one button that displays “Get”, and the last button that displays “Dump.”
   2. Put: Here are the requirements for the buttons the Put\* buttons

All Put\* buttons operate the same way except that they insert different values  
 with the same set of keys.

* + 1. When touched, it inserts ten <key, value> pairs into the storage by using my content provider’s insert().
    2. Each operation is preceded by a 3-second delay.
    3. Each touch of the button resets the sequence number to 0.
    4. The format of the <key, value> pairs is the following:
       1. Key: the sequence number represented as a string starting from 0 (e.g., “0”, “1”, “2”, etc.)
       2. Value: string for each button concatenated by each sequence number.
       3. For the button “Put1”, the values should be “Put10”, “Put11”, “Put12”, etc.
       4. For the button “Put2”, the values should be “Put20”, “Put21”, “Put22”, etc.
       5. For the button “Put3”, the values should be “Put30”, “Put31”, “Put32”, etc.
  1. Get
     1. This button retrieves ten keys, “0”, “1”, …, “9” with their corresponding values using your content provider’s query() interface. Again, each query should be preceded by a 3-second delay.
     2. Each retrieval displays the <key, value> pair retrieved on the screen. The format of the display is “<key, value>” as a string.
  2. Dump
     1. When touched, this button dumps all the <key, value> pairs stored in your local storage. Since you need to implement a distributed key-value storage based on Dynamo, each instance stores <key, value> pairs that belong to one partition as well as replicas from other partitions. This button should display all local <key, value> pairs on the screen.
     2. The order of <key, value> pairs you display does not matter as long as it shows all locally stored <key, value> pairs.
     3. Note that each local storage in this assignment should contain replicas as well. Obviously, the Dump button should also display these replicas when touched.