CS371m - Mobile Computing

Persistence - Web Based Storage
The Cloud ........

IBMCloud @IBMcloud · Aug 23
Hamsters: Cute, but dangerous. Trust us—you don't want to share your #cloud with a hamster.
Backend

• No clear definition of backend
• front end - user interface
• backend - data, server, programs the user does not interact with directly
• With 1,000,000s of mobile and web apps...
• rise of Backend as a Service (Baas)
• Sometimes MBaaS, M for mobile
Back End As a Service - May Provide:

- cloud storage of data
- integration with social networks
- push notifications
  - server initiates communication, not the client
- messaging and chat functions
- user management
- user analysis tools
- abstractions for dealing with the backend
MBaaS

Data

Push

Analytics

Social

Cloud Code

Hosting
Some Examples of MBAas

- Parse
- Amazon Web Services
- Google Cloud Platform
- Heroku
- PythonAnywhere
- Rackspace Cloud
- BaasBox (Open Source)
- Usergrid (Open Source)
Example of Using a MBaaS

- Parse
- www.parse.com
- various pricing models
- relatively easy to set up and use
- Going away 1/28/2017
Parse Set Up in AndroidStudio

1. request api key
2. Download Parse SDK
3. Unzip files
4. Create libs directory in app directory (select Project view)
5. Drag jar files to libs directory
6. add dependencies to gradle build file under app

like so:

```
dependencies {
    compile 'com.android.support:support-v4:18.0.0'
    compile 'com.parse.bolts:bolts-android:1.+'
    compile fileTree(dir: 'libs', include: ['Parse-*.jar'])
}
```

https://www.parse.com/apps/quickstart#parse_data/mobile/android/native/new
Testing Parse

• Add permissions to manifest to access network state and use internet

```xml
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"
<uses-permission android:name="android.permissionINTERNET" />
```

• initialize Parse in onCreate method

• keys for account and app

```java
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    Parse.initialize(this, "GACBq6Jwvf2PL7EI13IRpvav7GEU"
```
Testing Parse

• at the end of onCreate()
• create and send a test object to Parse

```java
    testParse();
```

```java
private void testParse() {
    ParseObject testObject = new ParseObject("TestObject");
    testObject.put("foo", "bar");
    testObject.saveInBackground();
}
```

• abstraction
  – handles doing this in the background, off the UI thread
Result of Test

Congrats! You saved your first object:

```
{ "id": "HQZcs4g5vp", "created_at": "2014-11-11T21:34:19Z", "updated_at": "2014-11-11T21:34:19Z", "foo": "bar" }
```

- **JSON**
  - JavaScript Object Notation
parseObject

private void testParse() {
    ParseObject testObject = new ParseObject("TestObject");
testObject.put("foo", "bar");
testObject.saveInBackground();
}

• Local representation of data (on the device) that can be saved and retrieved from the Parse
• String in constructor is class name
  – like a table in a data base
• put to add key - value pairs
  – String - Object
  – keys must be alphanumerics
  – like a column in the row
ParseObject

private void testParse() {
    ParseObject testObject = new ParseObject("TestObject");
    testObject.put("foo", "bar");
    testObject.saveInBackground();
}

- `saveInBackground` method saves object to Parse in a background thread
- multiple options for saving
  - `saveAll(List)`
  - `saveEventually()` - if server or network not available
  - `saveInBackground(SaveCallback)`
Parse and RandomArt

• add ability to save equations
• save to parse database
• allow multiple users to save equations
• functionality to display a random equation others liked
• up and down votes
onClick for Keep This

```java
clic void saveEquation(View v) {
    if(exp != null) {
        // should also check to ensure equation not already saved
        final int[] count = {0};

        ParseQuery<ParseObject> countQuery
            = ParseQuery.getQuery("ArtExpressionCount");

        countQuery.getFirstInBackground(new GetCallback<ParseObject>() {
            @Override
            public void done(ParseObject masterCount, ParseException e) {
                if(e == null) {
                    count[0] = masterCount.getInt("TheCount");
                    Log.d(TAG, "The Count via the master count object: ",
                    masterCount.increment("TheCount");
                    masterCount.saveInBackground();
```
onclick for Save Equation - cont.

```java
masterCount.saveInBackground();

ParseObject currentExpression = new ParseObject("ArtExpression");

currentExpression.put("equation", exp.toString());
currentExpression.put("votes", 1);
currentExpression.put("index", count[0]);
currentExpression.saveInBackground();
}
else {
    Log.d(TAG, "Unable to get count, not saving expression");
}
```
saveEquation

• Makes a query to get the number of rows in the expression table
  – uses another table with one row with one column (GACK, no auto increment function)
• callback method for completed query
• checks the count
• creates new ParseObject
• makes the index for this new expression the count (0 based indexing)
• saves the object and updates count object
Parse Dashboard

- Examine data uploaded from apps
demo Saving an Equation
Get Random Saved Art

• When user presses button pick a random saved expression and render that image
• We just save the expression so we must recreate image  
  – time vs. space trade off
• check count of values and pick random index
public void getRandomGoodArt(View v) {
    pickRandomExpression = false;

    ParseQuery<ParseObject> countQuery
            = ParseQuery.getQuery("ArtExpressionCount");

    countQuery.getFirstInBackground(new GetCallback<ParseObject>() {
        @Override
        public void done(ParseObject masterCount, ParseException e) {
            if (e == null) {
                int count = masterCount.getInt("TheCount");
                int randomIndex = r.nextInt(count);
                Log.d(TAG, "The Count via the master count object: " + count);

                ParseQuery<ParseObject> query
                        = ParseQuery.getQuery("ArtExpression");
                query.whereGreaterThanOrEqualTo("index", randomIndex);
                query.getFirstInBackground(setRandomExpressionFromQuery);
            } else {
                Log.d(TAG, "Unable to get count to get random expression");
            }
        }
    });
}
callback object

• pull out the String from the returned object and build expression based on equation

private GetCallback<ParseObject> setRandomExpressionFromQuery
    = new GetCallback<ParseObject>();

public void done(ParseObject object, ParseException e) {
    if (e == null) {
        String equation = object.getString("equation");
        exp = new RandomExpression(equation);
        // now draw it
        Log.d(TAG, "equation: " + equation);
        Log.d(TAG, "index of expression: " + object.getInt("index");
        new ArtTaskInner().execute(artImage.getWidth(), artImage.getHeight());
    } else {
        Log.d(TAG, "Unable to get the given random expression");
    }
}
<table>
<thead>
<tr>
<th>objectID</th>
<th>String</th>
<th>createdAt</th>
<th>updatedAt</th>
<th>equation</th>
<th>index of expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>EnfckzAUx4</td>
<td>Nov 11, 2014, 22:45</td>
<td>Nov 11, 2014, 22:45</td>
<td>xCCCSSyCySSSCySSSSMM_ASSSS</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Oi0XTrmtMc</td>
<td>Nov 11, 2014, 23:03</td>
<td>Nov 11, 2014, 23:03</td>
<td>yCQyCxCAMCSCCCxCCCSMS</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>
More Parse

• Includes capability to do local data store
  – save objects on device, save to cloud later
  – abstracts away a lot of the details
  – Kyle Norton: "Assume you WON'T be connected to the network."

• Parse objects meant to be "small"
  – less than 128 kb
  – not for images
  – Parse files for large pieces of data