How to Use this Template

1. Make a copy [File → Make a copy...]
2. Rename this file: “**Capstone_Stage1**”
3. Replace the text in green

Submission Instructions

1. After you’ve completed all the sections, download this document as a PDF [File → Download as PDF]
2. Create a new GitHub repo for the capstone. Name it “**Capstone Project**”
3. Add this document to your repo. Make sure it’s named “**Capstone_Stage1.pdf**”

GitHub Username: peet29

Thai News Update!

Description

Thai News Update! Bring latest news in Thailand to you in English so you don't have to look up news web by web anymore. You can read the news easy and share it to your friends.
News source

- Thai PBs

Intended User

This app is for anyone that want to read news easy.

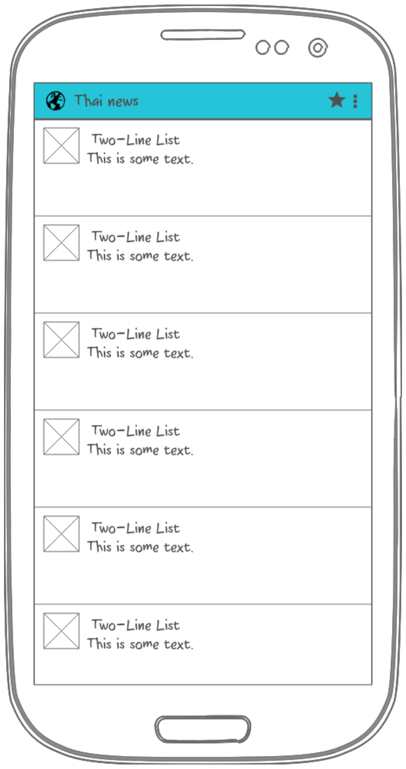
Features

- Display news in list
- Show news content in detail
- Keep news for read later
- Share news to other

User Interface Mocks

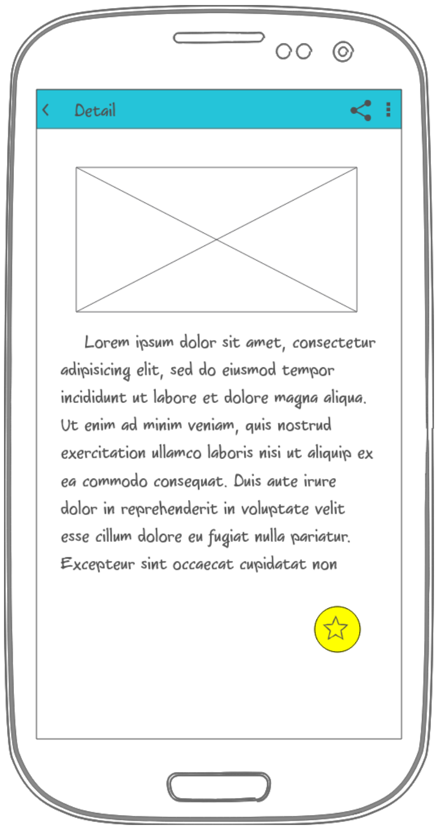
These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Photoshop or Balsamiq.

Screen 1



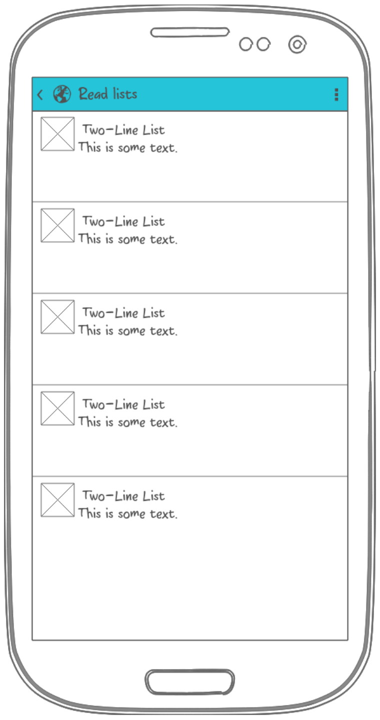
This is a main screen it will show news in list item user can scroll up and down.
If user hit list item it will open detail screen to show more of the news.

Screen 2


Detail
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non

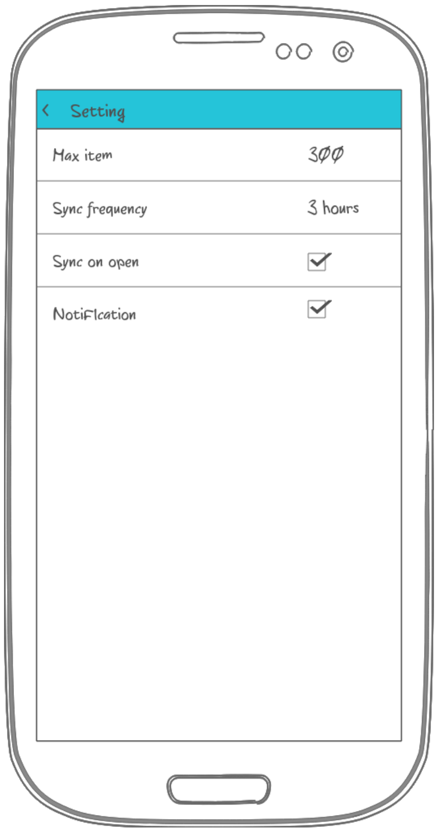
This is detail screen it show news in detail. User can share and save news in this screen.
If user have chrome it will open chrome custom tabs to display news instead.

Screen 3



This is a read lists screen it show news that user keep news that they want to keep for read it later in list. If user hit news in list it will open Detail screen

Screen 4



Max item 300
Sync frequency 3 hours
Sync on open ✓
NotiFIcation ✓

This is a setting screen user can change a setting of the app in this screen.

Key Considerations

How will your app handle data persistence?

I will build a Content Provider and link it to SQLite to keep to data.
It will have Shared Preferences to store all setting data

Describe any corner cases in the UX.

If user hit list item image in list item will get bigger as app move to detail screen.
After user hit keep news button it will show toast that this news is add to read list.
App will show why it can’t get data to show user.

Describe any libraries you'll be using and share your reasoning for including them.

- Glide for load image
- Volley for load Rss from server
- Schematic for create content provider

Next Steps: Required Tasks

This is the section where you can take the main features of your app (declared above) and decompose them into tangible technical tasks that you can complete incrementally until you have a finished app.

Task 1: Project Setup

Fist put all this in dependencies build.gradle in app level

- compile 'com.android.support:appcompat-v7:23.4.0'
- compile 'com.github.bumptech.glide:glide:3.6.1'
- compile 'com.github.bumptech.glide:volley-integration:1.3.1@aar'
- compile 'com.android.volley:volley:1.0.0'
- compile 'com.android.support:support-v4:23.4.0'
- compile 'com.android.support:design:23.4.0'
- compile 'com.android.support:cardview-v7:23.4.0'

For schematic put all this in build.gradle file

```
apply plugin: 'com.android.application'
apply plugin: 'com.neenbedankt.android-apt'

buildscript {
  repositories {
    mavenCentral()
  }

  dependencies {
    classpath 'com.android.tools.build:gradle:{latest-version}'
    classpath 'com.neenbedankt.gradle.plugins:android-apt:{latest-version}'
  }
}

dependencies {
  apt 'net.simonvt.schematic:schematic-compiler:{latest-version}'
  compile 'net.simonvt.schematic:schematic:{latest-version}'
}
```

I will use all of this libraries in this project

Glide

Glide is a an image loader it can fetch, decode, and display video stills, images and gif. I will use this to load all images in this app.

This is what Glide look like when use.

```
 Glide
    .with(myFragment)
    .load(url)
    .centerCrop()
    .placeholder(R.drawable.loading_spinner)
    .crossFade()
    .into(myImageView);
```

myFragment is a context
Url is a url of the image
myImageView in a view that Glide will load image to.

You can resize and crop the image with Glide to make it fit in view.
I will use volley to handle all http request

Volley

Volley is a Http library that makes networking for Android apps easier .

This is a sample of volley

```
final TextView mTextView = (TextView) findViewById(R.id.text);
...

// Instantiate the RequestQueue.
RequestQueue queue = Volley.newRequestQueue(this);
```

```
String url ="http://www.google.com";

// Request a string response from the provided URL.
StringRequest stringRequest = new StringRequest(Request.Method.GET, url,
            new Response.Listener<String>() {
    @Override
    public void onResponse(String response) {
        // Display the first 500 characters of the response string.
        mTextView.setText("Response is: "+ response.substring(0,500));
    }
}, new Response.ErrorListener() {
    @Override
    public void onErrorResponse(VolleyError error) {
        mTextView.setText("That didn't work!");
    }
});
// Add the request to the RequestQueue.
queue.add(stringRequest);
```

Schematic

Schematic is a library that Automatically generate ContentProviders

First create a class that contains the columns of a database table.

```
public interface ListColumns {

  @DataType(INTEGER) @PrimaryKey @AutoIncrement String _ID = "_id";

  @DataType(TEXT) @NotNull String TITLE = "title";
}
```

Then create a database that uses this column

```
@Database(version = NotesDatabase.VERSION)
public final class NotesDatabase {

  public static final int VERSION = 1;

  @Table(ListColumns.class) public static final String LISTS = "lists";
}
```

And finally define a ContentProvider

```
@ContentProvider(authority = NotesProvider.AUTHORITY, database = NotesDatabase.class)
```

```
public final class NotesProvider {

  public static final String AUTHORITY = "net.simonvt.schematic.sample.NotesProvider";

  @TableEndpoint(table = NotesDatabase.LISTS) public static class Lists {

    @ContentUri(
        path = "lists",
        type = "vnd.android.cursor.dir/list",
        defaultSort = ListColumns.TITLE + " ASC")
    public static final Uri LISTS = Uri.parse("content://" + AUTHORITY + "/lists");
  }
```

Table column annotations

```
@AutoIncrement
@DataType
@DefaultValue
@NotNull
@PrimaryKey
@References
@Unique
```

Defining an Uri

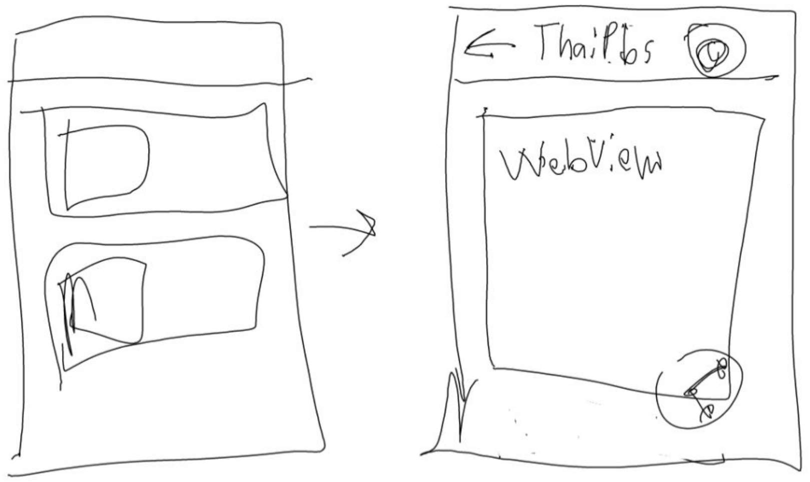
The @ContentUri annotation is used when the Uri does not change.

```
@ContentUri(
  path = "lists",
  type = "vnd.android.cursor.dir/list",
  defaultSort = ListColumns.TITLE + " ASC")
public static final Uri LISTS = Uri.parse("content://" + AUTHORITY + "/lists");
```

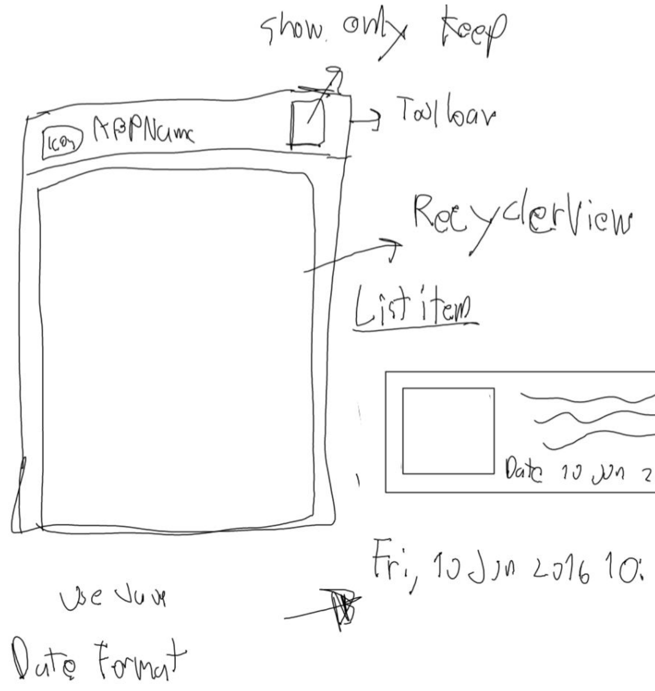
If the Uri is created based on some value, e.g. an id, @InexactContentUri is used.

```
@InexactContentUri(
  path = Path.LISTS + "/#",
  name = "LIST_ID",
  type = "vnd.android.cursor.item/list",
  whereColumn = ListColumns._ID,
  pathSegment = 1)
public static Uri withId(long id) {
  return Uri.parse("content://" + AUTHORITY + "/lists/" + id);
}
```

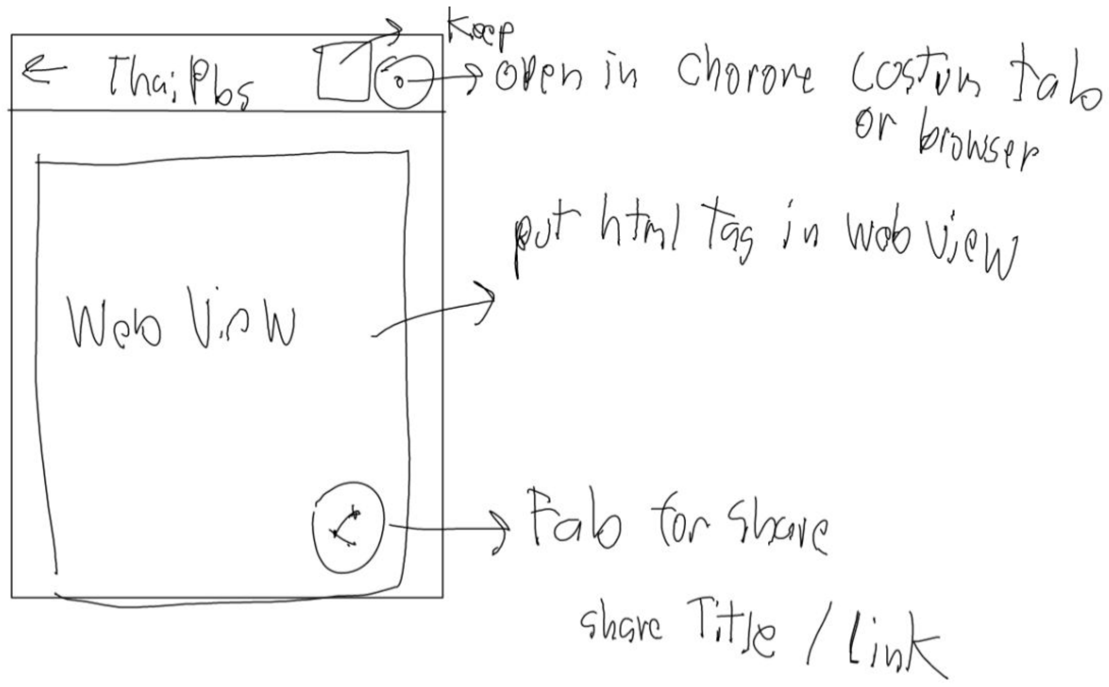
Task 2: Implement UI for Each Activity and Fragment



For that MainActivity it it will only have a toolbar and a Fragment.
In that Fragment will have only Recyclerview.



In list item will have CardView , ImageView and TextView.



This is a DetailActivity. It will have toolbar and Fab for a action and in main area will have a WebVew that will display news.

Task 3: Create UI

Create UI for all Activity

- Create layout for MainActivity
- Create layout for mainFragment
- Create layout for list item
- Create layout for DetailActivity
- Create layout for DetailFragment
- Create layout for SettingActivity

Task 4: Implement syncadapter

Implement syncadapter and find a way to use volley in it.
Check a

- Build syncadapter with volley
- Parsing XML Data
- Check data that volley download

Task 5: Create Content Provider

Use schematic to create content provider and put data from volley in it.

- Create Content Provider
- Save Data to ContentProvider

Task 6: Implement RecyclerView

Create RecyclerView and use loader to get data from content Provider

- Create RecyclerView
- Implement loader
- Check if there is data or not if not make a sync
- Show text in view if there is no data to show

Task 7: Implement DetailActivity

In DetailActivity will have fragment that have webview that will have data from content provider

- Implement loader
- Load data to webview

Task 8: Implement Share

Make a share button to share link to news.

- Create share button

Task 9: Implement Notifications

Create notifications to show when app fetch a new data

- Create Notifications in syncadapter
- When hit notifications will open DetailActivity

Task 10: Implement SettingsActivity

Create SettingsActivity to set setting

- Create setting
- Store setting data in Shared Preferences

Task 11:Implement Clear item

It a method that will delete oldest item if items are over Maxitem but will keep item in read llist.

- Create method to delete oldest item

Task 12: Implement read list

Read list is a list that keep news to read later.

- Create a read list button in MainActivity
- Create save news in DetailActivity
- Make button in MainActivity to open ReadlistsActivity
- Make button in DetailActivity save news in list

Add as many tasks as you need to complete your app.

Submission Instructions

1. After you've completed all the sections, download this document as a PDF [File → Download as PDF]
2. Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
3. Add this document to your repo. Make sure it's named "**Capstone_Stage1.pdf**"