# Article - Fragments

This Article is created by Casper Gross Larsen and Jens Egeberg Rasmussen. By reading this article you will learn about Fragments and the use of these in our own little school project. But before we can dive into What fragments are, we are going to talk a little about where fragments are used: in Android-App-Development.

Ano 2018 we are quite sure that everyone knows what an app is. App stands for application and is used as both web-app, desktop-app and mobile-apps. Mobile-apps is where we have been working via android studio, which is a tool for creating android-mobile-apps.

When you, as a user, open an app, what you see is the apps interface. This interface is probably filled buttons, text and other gadgets. This is an interface for android studio code. And this interface is presented by what is called an activity.

An activity is considered one of the fundamental building blocks of app on an android platform. This is where the user interacts with the app, and also serves as a navigator within an app (the back button, and Recent button serves as examples by this). The activity has a lifecycle in which, when the user clicks around on the app, the state of the activity changes within the same lifecycle. When the app closes the lifecycle ends, but data can still be stored. For instance, if you are watching a video on YouTube on your phone, and you pause and leave YouTube the lifecycle will save the state, from which you left the app and will resume next time you enter the app and be ready to continue from where you left off.

Another way to navigate through your activity is by the use of fragments.

A fragment is usually used as part of an activity's user interface and contributes its own layout (User Interface) to the activity. To provide a layout for a fragment, you must implement the onCreateView() call-back method, which the Android system calls when it's time for the fragment to draw its layout. A Fragment can be called a sub Activity.

Fragments are used to create more dynamic and flexible UI designs on larger screens, such as tablets. The larger screen makes more room to combine and interchange UI components (fragments). Fragments allow such designs without the need to manage and save the complex changes in the activity. By dividing the Activity into fragments, you will be able to modify the activity's appearance at runtime and preserve and save changes in the back, because it is all in the same activity.

What is the purpose of Fragments?

For example, a news application can use one fragment to show a list of articles on the left, and another fragment to display an article on the right—both. Fragments appear in one activity, side by side, and each fragment has its own set of lifecycle callback methods and handle their own user input events. Thus, instead of using one activity to select an article, and another activity to read the article, the user can select an article and read it all within the same activity.

Each fragment should be designed as a modular and reusable activity component. That is because a fragment defines its own layout (User interface) and its own behaviour within its own lifecycle and call-backs (code methods used to change the state). Each fragment can be included in multiple activities, so the design should be reusable.

This is especially important, because a modular fragment allows you to change your fragment combinations for different screen sizes. When designing your application to support both tablets and handsets, you can reuse your fragments in different layout configurations to optimize the user experience, based on the available screen space. For example, on a handset, it might be necessary to separate fragments to provide a single-pane UI when more than one cannot fit within the same activity.

Alternatively, in Android Development, you can use the Activity. The activity is able to navigate throughout the app as well and multiple activities can be swapped, however only one activity can be viewed at a time. while multiple fragments can be viewed from within an activity.

The main reason to use Fragments are for the backstack and lifecycle features. Otherwise, custom views are more light weight and simpler to implement. By using fragments, we are able to save the state, and benefit from lifecycle features. Another great reason for the use of fragments is its <----> ????

-------------------------------------

Summing up our school project: <br> <br>

We are creating a mobile android app in which you can create and login as a user. The user can then choose from a wide variety of recipes or add recipes for other people to enjoy. First of all, a login system is created.

Then of course there is the adding and viewing of recipes. The navigation is created with fragments. (picture below)

![alt text](https://github.com/J-Egeberg/BallBoiV5/blob/master/Menu.png)

Each fragment is placed in its own class, so each fragment can have the methods and functions that the fragment requires to work as intended. For instance, the fragment below is the login fragment and right now it contains a method that when you click the login button it renders the fragment Menu which has its own layout file.

![alt text](https://github.com/J-Egeberg/BallBoiV5/blob/master/Fragment.png)

Each fragment has its own layout file which is what the user can see and interact with. So, f. ex. the Menu fragment has a menu layout file and so on. So, whenever each fragment is called the layout file is rendered. (picture below of layout file).

![alt text](https://github.com/J-Egeberg/BallBoiV5/blob/master/Layout.png)

In our school project we have decided to use fragments for navigational purposes. The navigation (fragments) is all happening within the Main Activity and is swapped when clicking around on the app by our class SectionStatePagerAdapter (look photo below). We are doing this, because we want to save info in the overall activity from a logged in user. Saving the data allows us to save recipes from a specific user, and therefore we can use his name as a key for each recipe and furthermore create a user binding to each recipe.

![alt text](https://github.com/J-Egeberg/BallBoiV5/blob/master/FragmentManager.png)

The setup with fragments has been smooth so far. It's a fantastic way of making the navigation in the project, very smooth. It's simplistic and it's very easy to manage and handle the various classes in our application. We decided to make use of fragments after learning about them in class - It's simply a better way of navigating within a project.