Tutorial 4

Have a go yourself!!!

Build an application, which uses a web component to run the Check-in command. It can log either an IMEI or student number to the server.
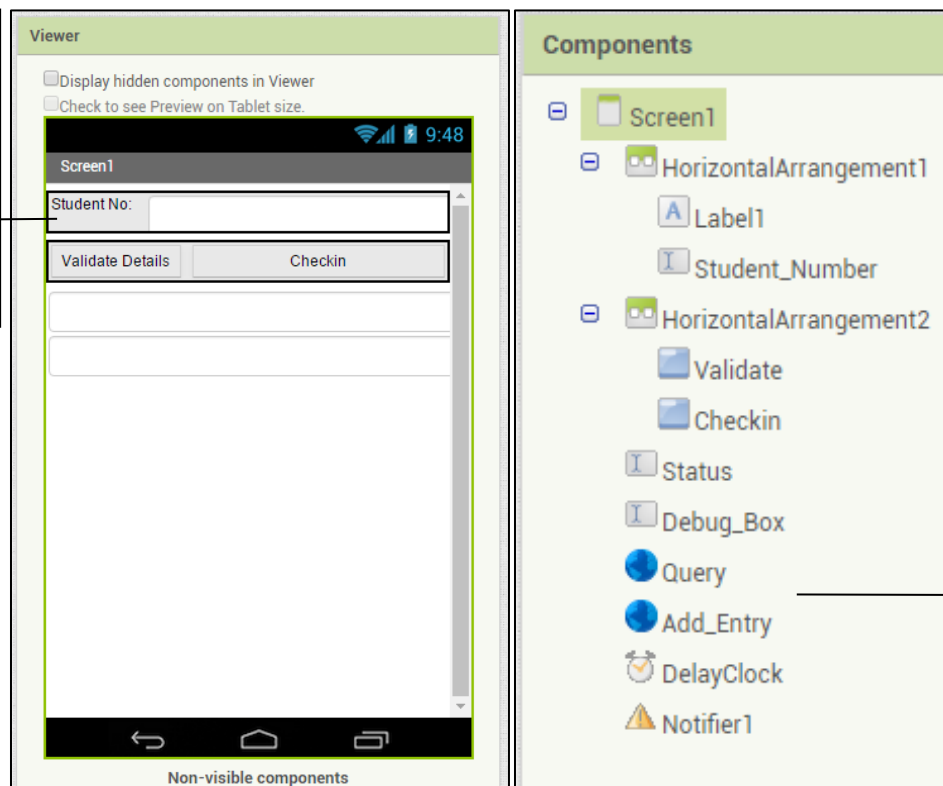
1. Add a Check-in button to the application. When this button is clicked, build the string which will be sent to the server.

2. Add another web component which can be used to run the Query command. Add a query button, so that when the button is clicked, the server will be queried about the existence of specific data.

    Hint: for each web component, use the GotText event and ensure it has a conditional statement, to let you know if the call was successful or not.

    Hint: if using the student number, you may wish to include some validation that the student number is correct (for example, check the length of the student number, check that the student number starts with a lowercase c, etc).

Designer

This example uses a text box to obtain a student number. It then has buttons "Validate" and "Checkin" which will be activated depending on which activity is necessary.

This example uses one web component for **Query,** and the other web component for **Checkin** (shown here as "AddEntry")

Blocks


```
when Screen1 .Initialize
do   call reset_screen
```

```
to reset_screen
do  set Student_Number . Hint to   " CXXXXXXX "
    set Student_Number . Text to   " "
    set Student_Number . Enabled to   true
    set Validate . Enabled to   true
    set Checkin . Enabled to   false
```

Here, the application uses a validation button click to build the query and send it to the server. Note that this uses the Query web component to handle the query communication.
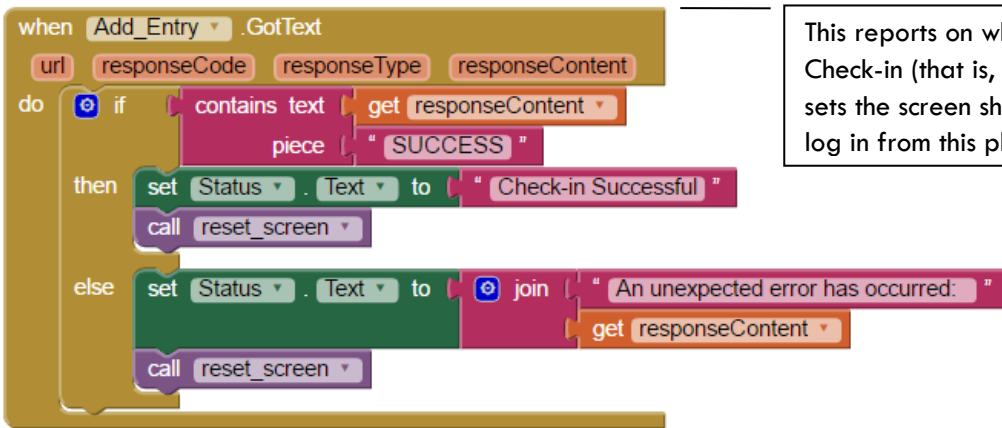
```
when Validate .Click
do  set Query . SaveResponse to   false
    set Query . Url to   join  " https://engg1500.newcastle.edu.au/ENGG1500DemoServer/API? "
                              " command=query "
                              " &group=GROUP1 "
                              " &uid= "
                              Student_Number . Text
    call Query .Get
```

```
when Query .GotText
url responseCode responseType responseContent
do  if   contains text   get responseContent
           piece   " SUCCESS "
    then set Status . Text to   " You are able to login! "
         set Checkin . Enabled to   true
    else set Status . Text to   " This student has already logged in. You cannot login again. "
         set Checkin . Enabled to   false
```

If the query is successful, the program can move onto the Check-in feature. This is allowed by enabling the "Checkin" button.

```
when Checkin .Click
do  set Add_Entry . SaveResponse to   false
    set Add_Entry . Url to   join  " https://engg1500.newcastl
                              " command=checkin "
                              " &group=GROUP1 "
                              " &uid= "
                              Student_Number . Text
    call Add_Entry .Get
```

When the Checkin button is clicked, the command to checkin is built and sent to the server. Note that this time, the "Add_Entry" web_component is to handle the checkin commands.
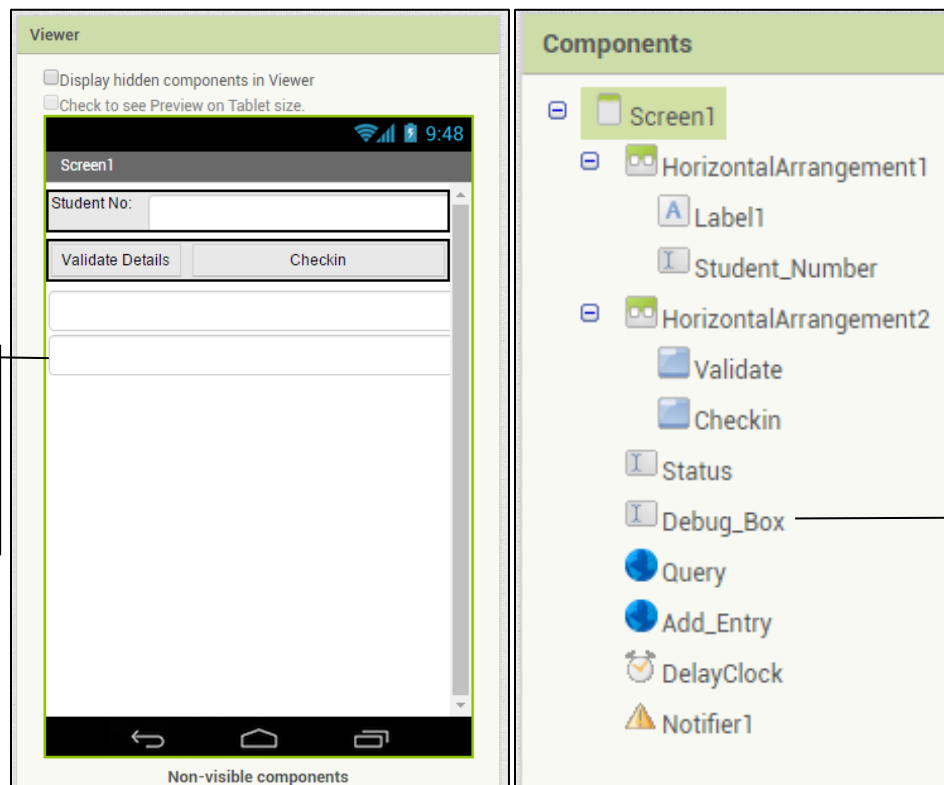
when Add_Entry .GotText
 url responseCode responseType responseContent
 do ☼ if ⸨ contains text [get responseContent ▾]
              piece [" SUCCESS "]
    then set Status ▾ . Text ▾ to [" Check-in Successful "]
          call reset_screen ▾
    else set Status ▾ . Text ▾ to [☼ join [" An unexpected error has occurred: "]
                                             [get responseContent ▾]]
          call reset_screen ▾

This reports on whether the student could Check-in (that is, was it successful). It then re-sets the screen should another student wish to log in from this phone.

Note that the example above is intentionally limited. This example is only to demonstrate the use of the two web-components in the one application. The application you and your group design will require more logic so that any user input (if used) is validated locally (i.e. within the application), as well as incorporating more features which have been demonstrated in tutorials, or features found using your own research.

Useful Development Feature

During development, it is possible you may wish to include a debugging method, which will help you determine the cause of any unexpected application behavior.
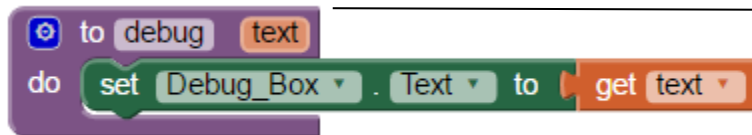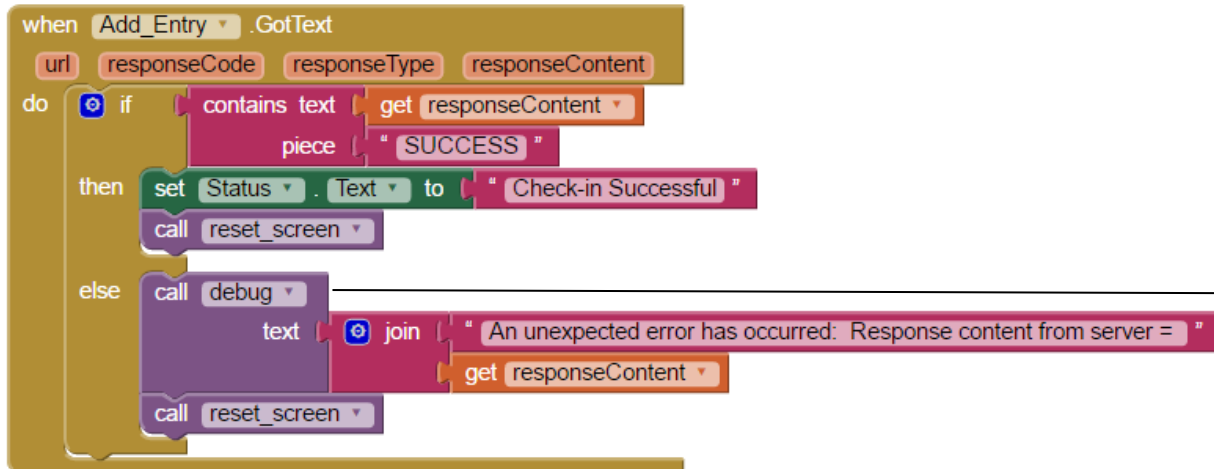
Designer

A text box at the bottom of the screen can be populated with debug information if/when required.

Note that this text box should be removed before the application is submitted for marking.

Blocks

A debug method can be made which takes some text as input. That text will be displayed to screen.

This method can be called from any part of your program, where unexpected behaviour occurs. This will give you more information, so that you can determine the cause of the error.