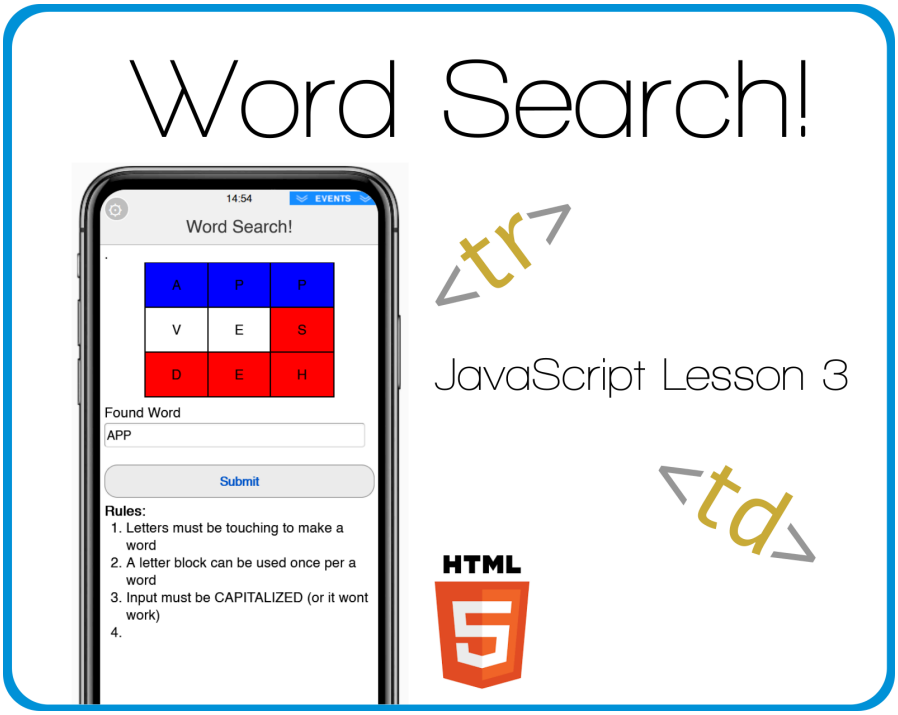
JavaScript Lesson 3: Word Search!

Overview

Introduction

In this lesson we are going to use **HTML, CSS** and **JavaScript** to create a word search game that highlights words when the user gets it correct!



Prerequisites

We recommend finishing the following lessons before starting this one if you are new to the App Builder:

- **App Builder Basics**
- **JavaScript Lesson 2**

Learning Objectives

By the end of this lesson:

- **All students** should have built their own **Word Search**

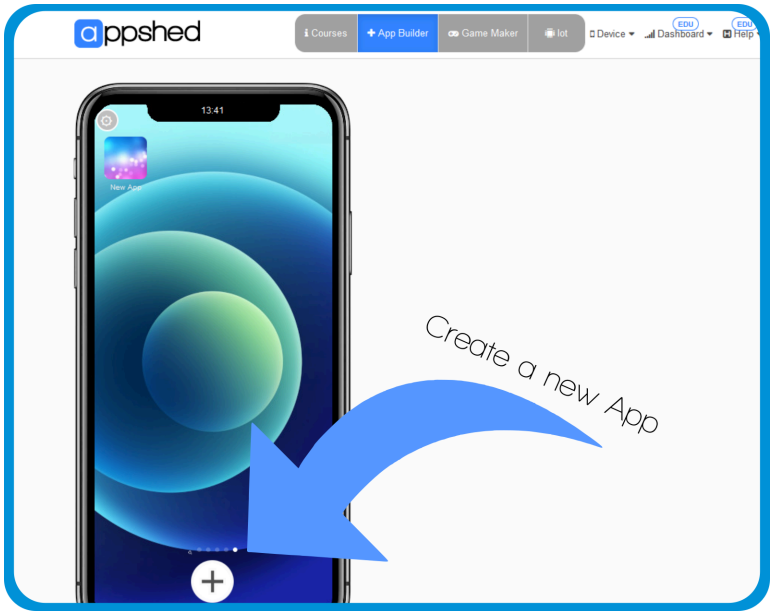
- **Most students** will **change** the Word Search **JavaScript** and **HTML**
- **Some students** will build the **more advanced** version of this project

Get the App Ready

New App

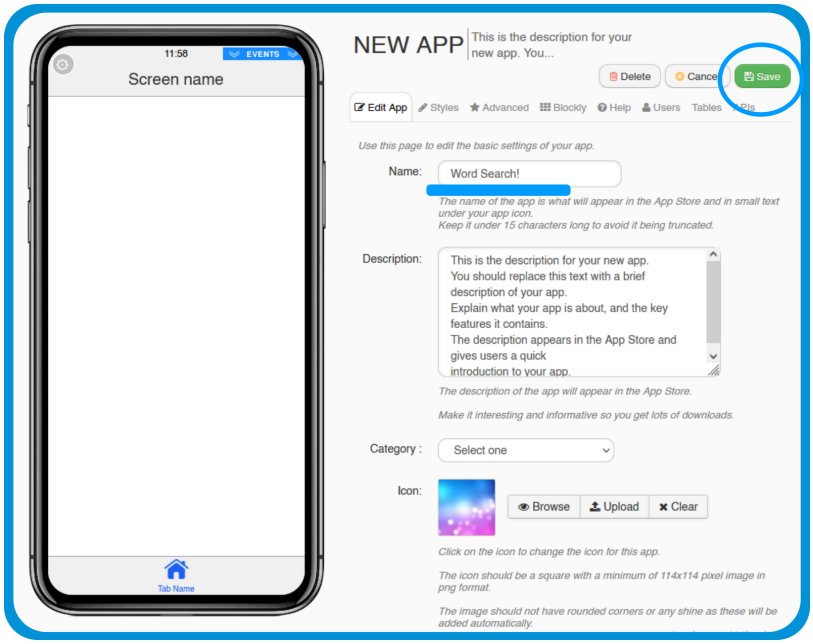
As always, the first thing we need to do is create a new app for our project and give it a name. Make sure you're **logged into** AppShed and navigate to the App Builder.

- **Click** the **Plus** button



Now with our new app created let's give it a name

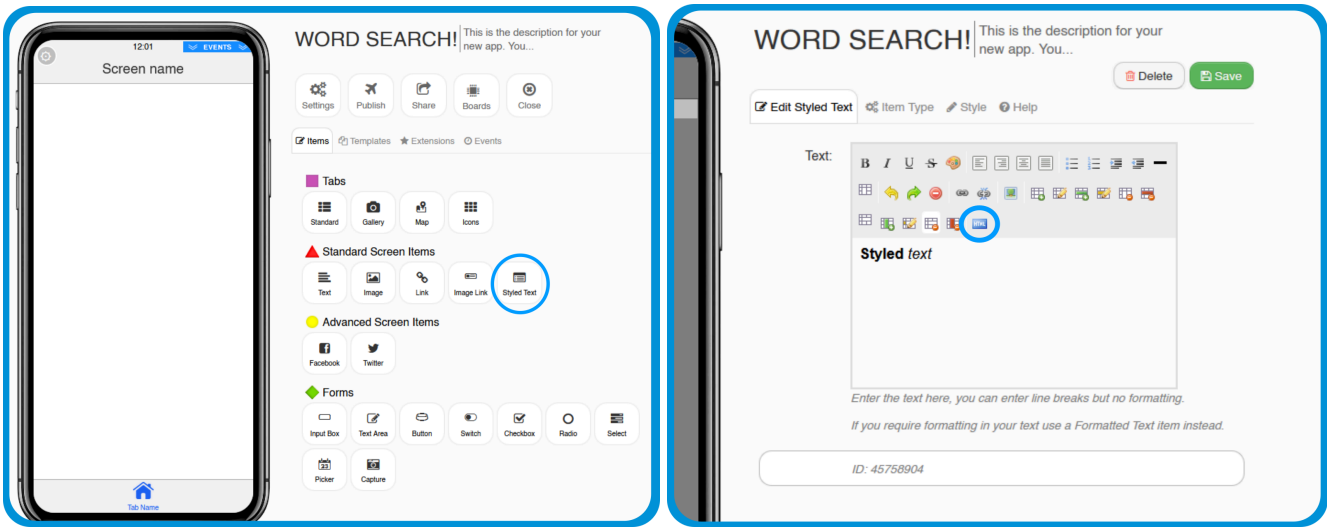
- **Click Settings**
- In the text box named **Name** type **Word Search!**
- **Click** the **Green Save** button



Creating a Table

The letters for our word search will be contained in a 3x3 table which we will add to our app using the <table> tag in HTML.

- Click on **Styled Text** under **Standard Screen Items**
- Click on the **HTML** button to swap to the HTML editor (the last button located to mid/to the right)
- **Delete** anything there already



Here is where we will put our HTML to create a table, we want a 3x3 table that looks like this:

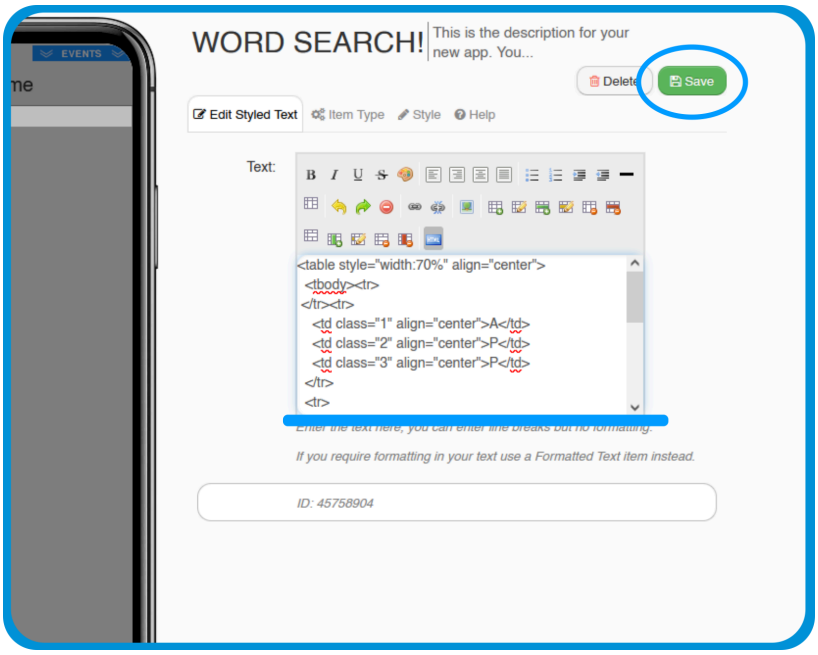
A	P	P
V	E	S
D	E	H

(Later on you can add your own letter but for now use the ones we have to make sure your code works!)

- In the HTML editor **paste** the following code

```
<table style="width:70%" align="center">
  <tbody><tr>
 </tr><tr>
      <td class="1" align="center">A</td>
      <td class="2" align="center">P</td>
      <td class="3" align="center">P</td>
  </tr>
  <tr>
      <td class="4" align="center">V</td>
      <td class="5" align="center">E</td>
      <td class="6" align="center">S</td>
  </tr>
  <tr>
      <td class="7" align="center">D</td>
      <td class="8" align="center">E</td>
      <td class="9" align="center">H</td>
  </tr>
</tbody></table>
```

- Click **Save**



Breaking Down the HTML

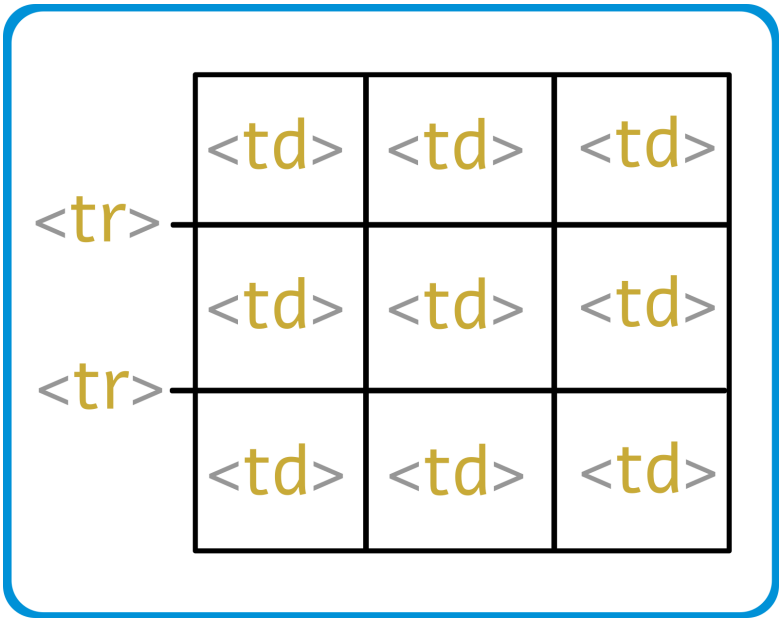
See if you can figure out what's going on with this quick hint:

`<tr>` = Table Row
`<td>` = Table Data

What we have just done is created a table with three rows (3x `<tr>`) and three columns (each row has 3x `<td>`) and finally we put a letter into each column.

The most **important** thing we did however is **give each cell in our table a class name**, we did this by putting `class="x"` in each `<td>` tag, like this `<td class="1">A</td>`**.**

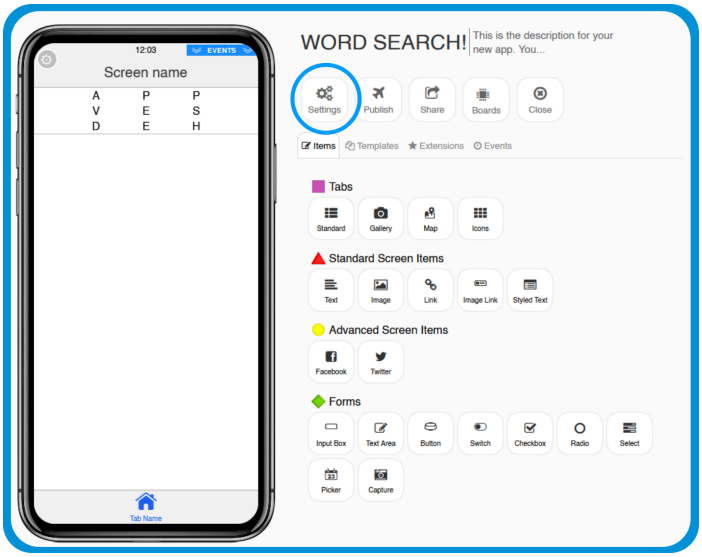
You should notice that **each letter has its own class name** from 1-9. So **A = Class 1** and **H = Class 9**. This is going to help us later when we start coding JavaScript.

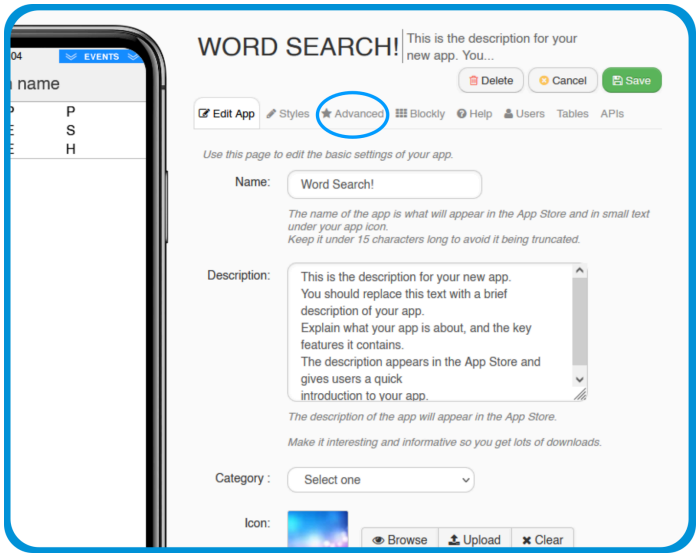


A little CSS

Our table isn't quite looking like a table yet, the letters are in a grid which is correct but there are no defining borders! We will add these borders with CSS.

- Click on **Settings**
- Click **Advanced**





Here we can see the Custom CSS editor, this is where we can apply some CSS to our table.

- In the Custom CSS editor **type**:

```
td {
  border: 2px solid black;
  padding: 15px;
}
```

- Click **Save**

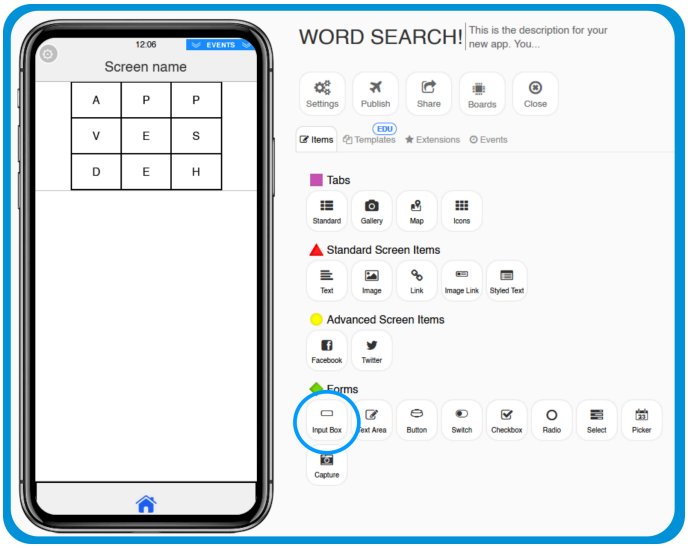
What we have just done with the CSS is apply a border to all `<td>` tags in our app and then we put a padding of 15 pixels inside the `<td>` or table cell to make them more square.

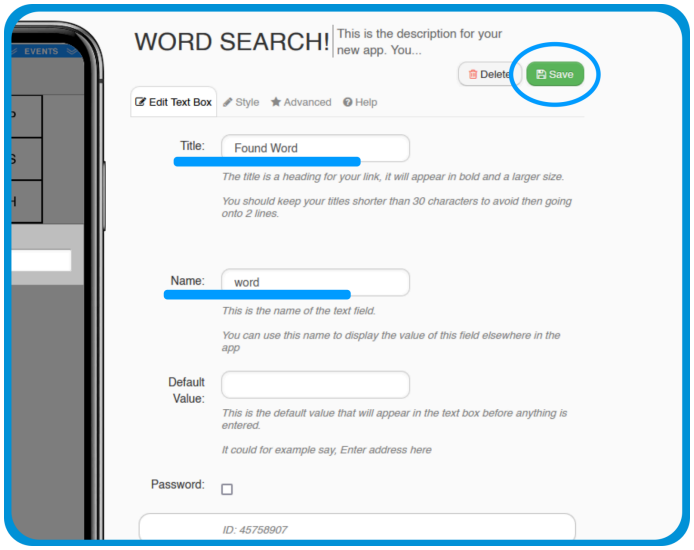
If you would like to learn more about creating tables with HTML and styling them with CSS check out the HTML Table example in the HTML & CSS Reference Section.

The Other Stuff

Now let's quickly add the other stuff so we can get to the coding! We will need to add an input box where the user can input words they find, a submit button which will house the code used to check if the word is valid and finally a text item with some rules.

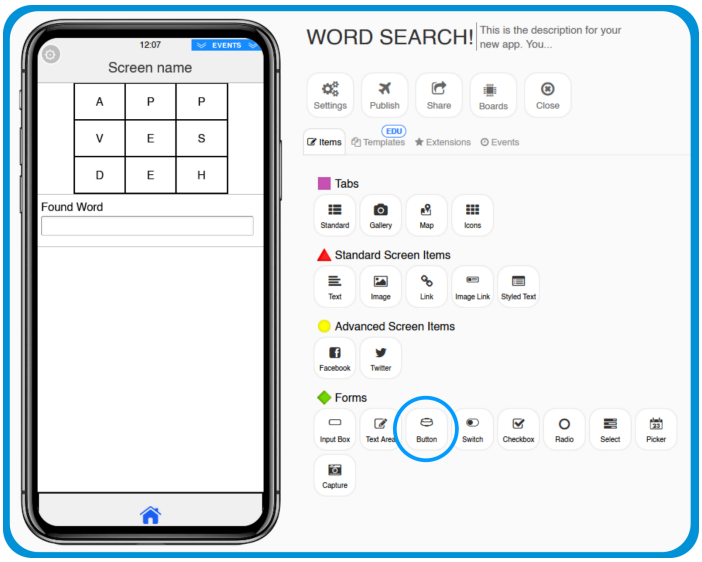
- Click on **Input Box** under **Forms**
- In the **Title** type: **Found Word**
- In the **Name** type: **word**
- Click **Save**

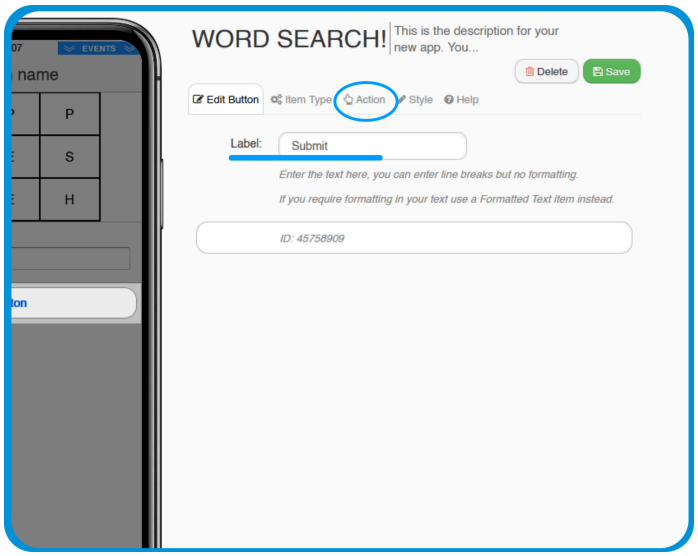


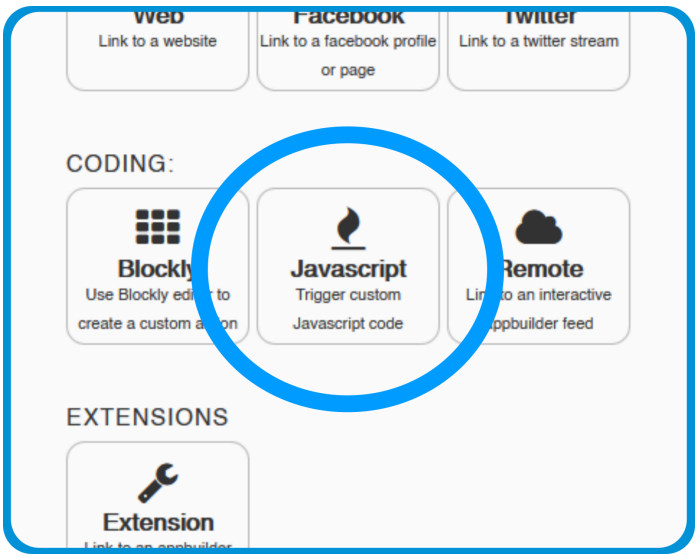


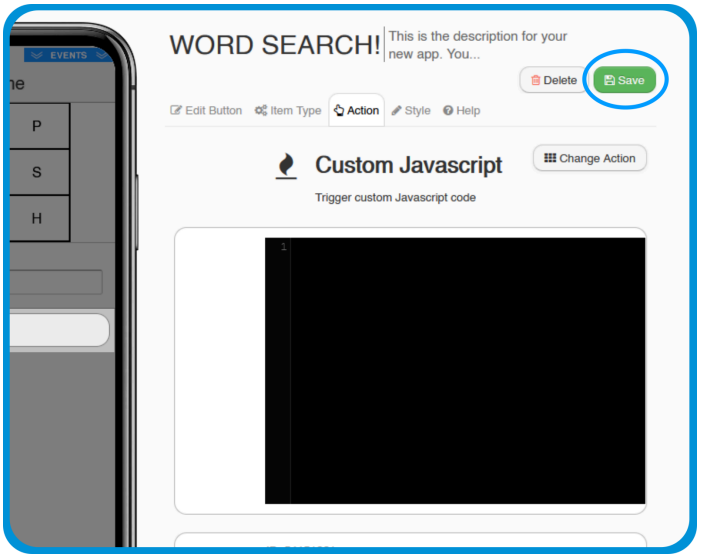
Remember that the Name of the input box is the variable name that we will use in coding later.

- Click on **Button** under **Forms**
- In the **Label** type: **Submit**
- Click **Action**
- Click **Change Action**
- Click **JavaScript**
- Click **Save**









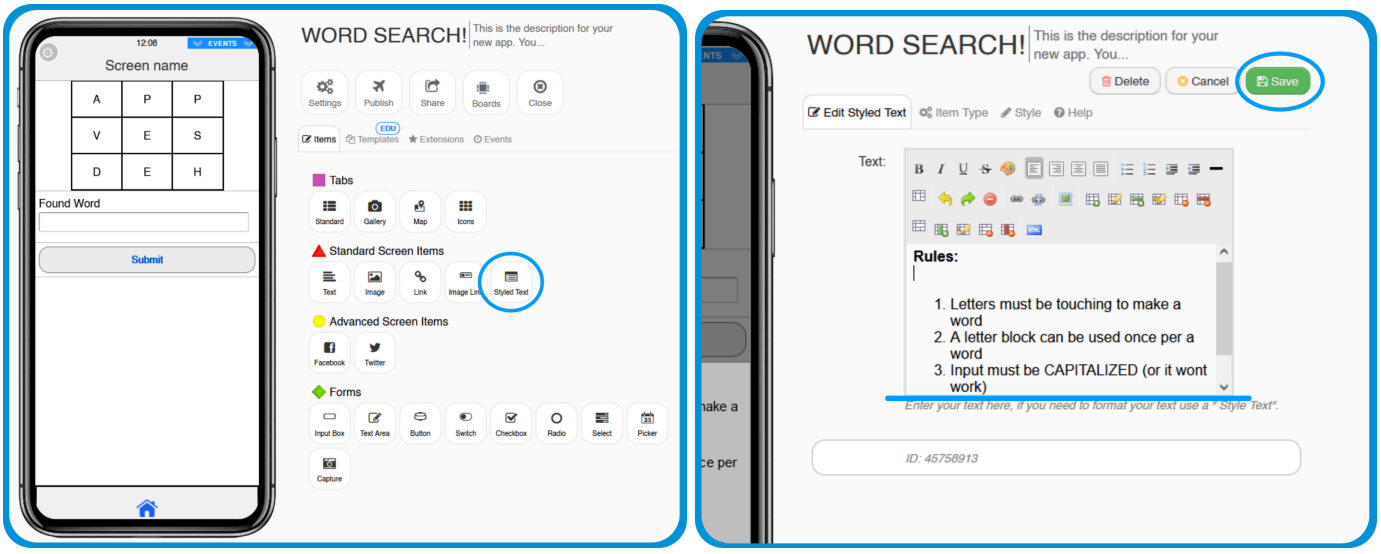
We haven't added our JavaScript yet, we are just getting our button ready for when we do it in the next step.

- Click on **Styled Text**
- Type the following:

Rules:

1. Letters must be touching to make a word
2. A letter block can be used once per a word
3. Input must be CAPITALIZED (or it wont work)

- Click **Save**

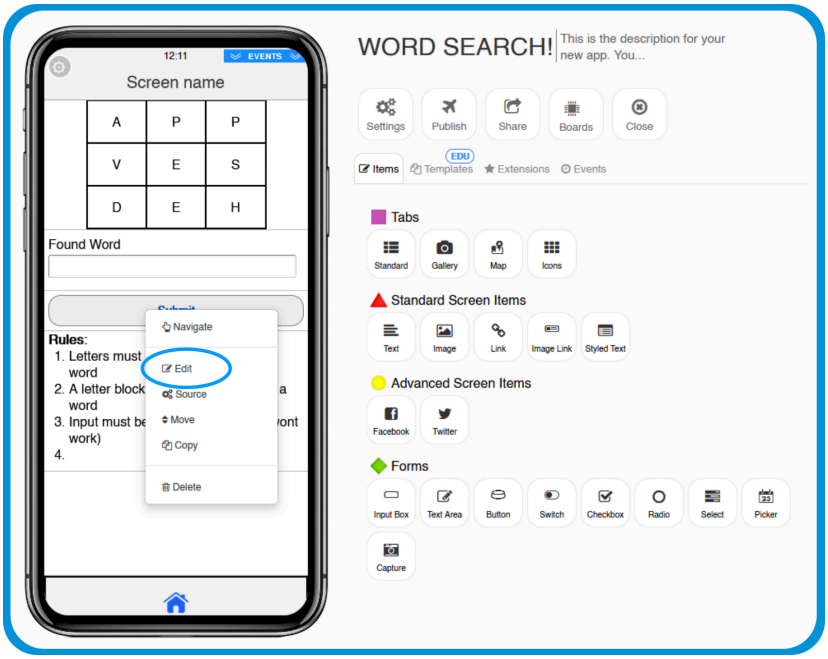


Coding

Edit the Button

Let's now get back to the JavaScript coding editor on our button.

- Click on the **button we just added**
- Click **Edit**
- Click **Action**



Now we can start coding!

The If Statement

We should be nice and comfortable with the IF Statement by now as we are going to use it the exact same way we have in previous lessons, in order to evaluate the status of our input box.

- Type the following into the JavaScript Editor:

```
if (app.getVariable('word') == 'APPSHED') {
    //output
}
```

In the code above we are stating that if the user types APPSHED something will happen in the output. We will come back to our condition in a moment but for now let's set up all of our If statements, we will need an IF statement for each available word in our table.

(There are actually 49 words in this simple 3v3 table, we will do four together, the rest is for you to try and find!)

We can add our next IF statement as an ELSE IF statement, this is used to check another condition if the first one returns false.

- Type the following into the JavaScript Editor

```
if (app.getVariable('word') == 'APPSHED') {
    //output
}
else if (app.getVariable('word') == 'APP') {
    //output
}
else if (app.getVariable('word') == 'DEAP') {
    //output
}
else if (app.getVariable('word') == 'PAVED') {
    //output
} else {
    alert('nope, well maybe, we did not add them all');
}
```

As you can see we can add as many ELSE IF Statements as we want. Once we have added all that we want to, we can end it with an ELSE statement. The ELSE statement checks if all other statements are false and if they are it is activated, in its output we add an alert which will tell the user they didn't input a correct (or known) word.

The Output

Now that we have added all of our conditional statements it's time to add our output. What we want our output to do is to highlight the relevant cells of the table depending on which word the user types, luckily we gave each cell a class name earlier making it easy for us now!

We will use **jQuery** to get the relevant cells and then a **jQuery** method called **.css** to apply CSS to them. The CSS we apply will simply be `background-color: blue;`

First Word

Let's start with our first word, APPSHED. When the user type this and clicks submit the 1st, 2nd, 3rd, 6th, 7th, 8th and 9th cell should light up. See the little cheat sheet bellow if this is confusing

(1) A	(2) P	(3) P
(4) V	(5) E	(6) S
(7) D	(8) E	(9) H

To do this our code will look like this:

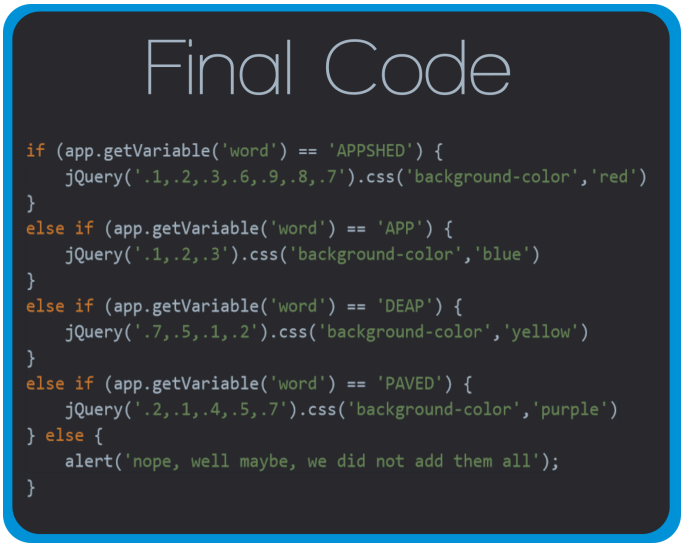
```
jQuery('.1,.2,.3,.6,.7,.8,.9').css('background-color','red')
```

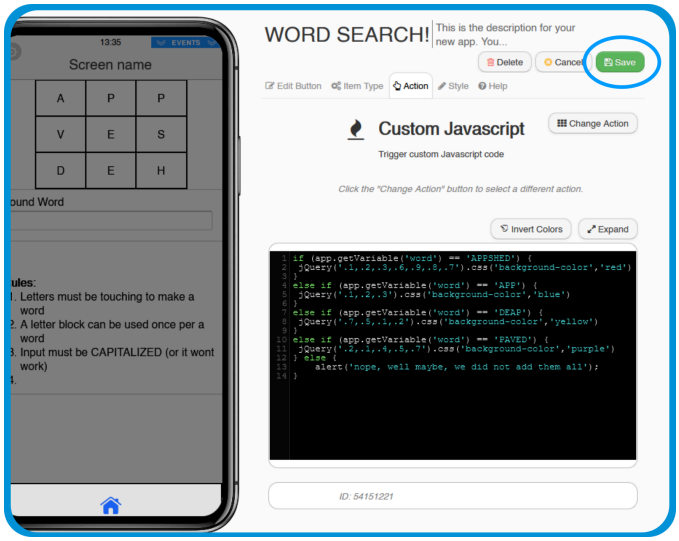
If this is what the code for APPSHED would look like, what would the code for the rest look like, see if you can figure it out yourself!

Here is what your final code should look like:

```
if (app.getVariable('word') == 'APPSHED') {
    jQuery('.1,.2,.3,.6,.9,.8,.7').css('background-color','red')
}
else if (app.getVariable('word') == 'APP') {
    jQuery('.1,.2,.3').css('background-color','blue')
}
else if (app.getVariable('word') == 'DEAP') {
    jQuery('.7,.5,.1,.2').css('background-color','yellow')
}
else if (app.getVariable('word') == 'PAVED') {
    jQuery('.2,.1,.4,.5,.7').css('background-color','purple')
} else {
    alert('nope, well maybe, we did not add them all');
}
```

- Make sure you're code **matches** the **above code**
- Click **Save**

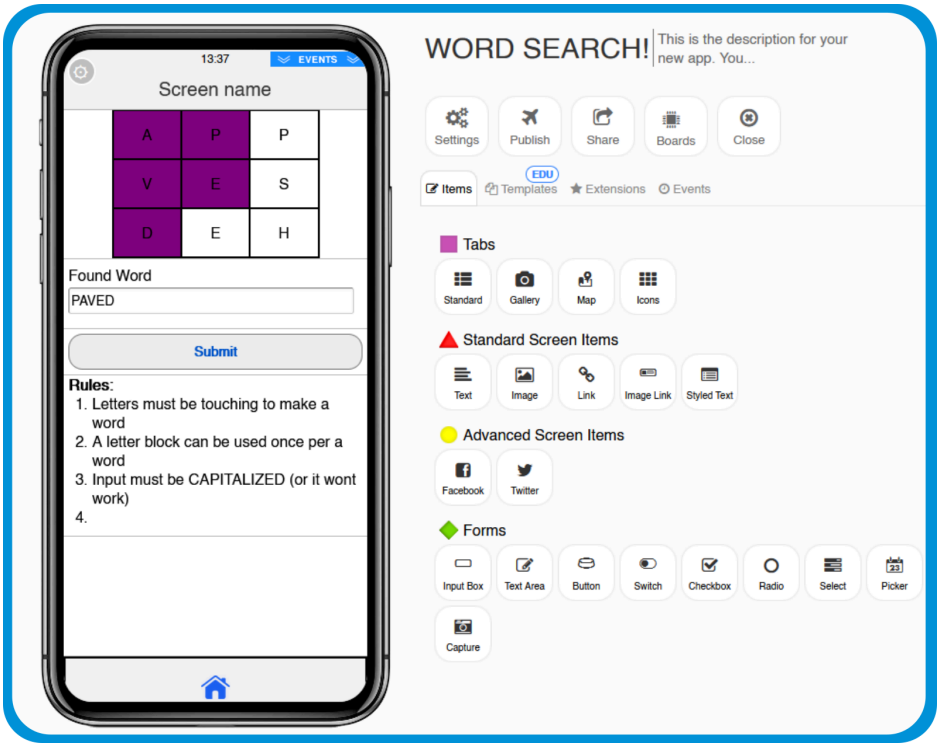




Give it a Test

Let's now give it a test.

- Type **PAVED** into the **input box**
- Click on the **Submit button**
- Click **Navigate**



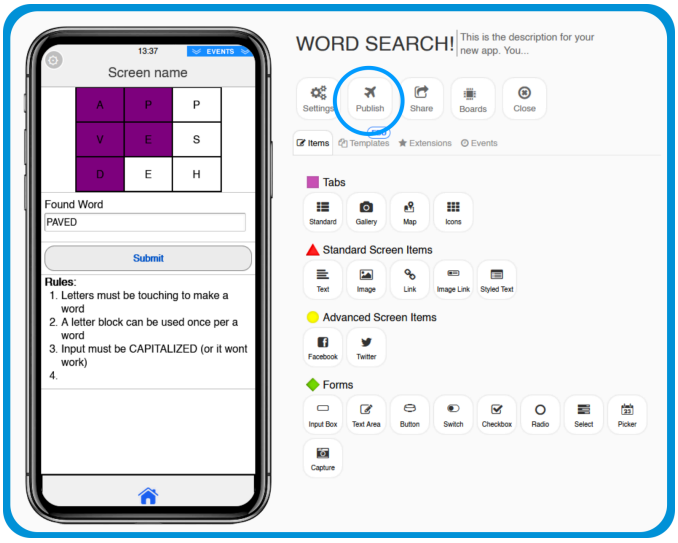
We should see all the letters of PAVED get highlighted in purple! Now we have only added 4 words out of a possible 49, see if you can figure out all the words and add them to your code.

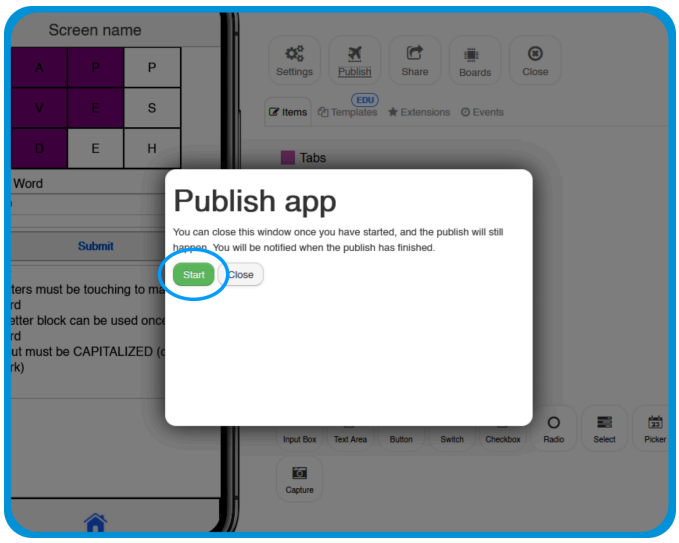
Conclusion

Publish the App

Like with all apps on AppShed we can publish it and **view it on our phones** or even **share it.**

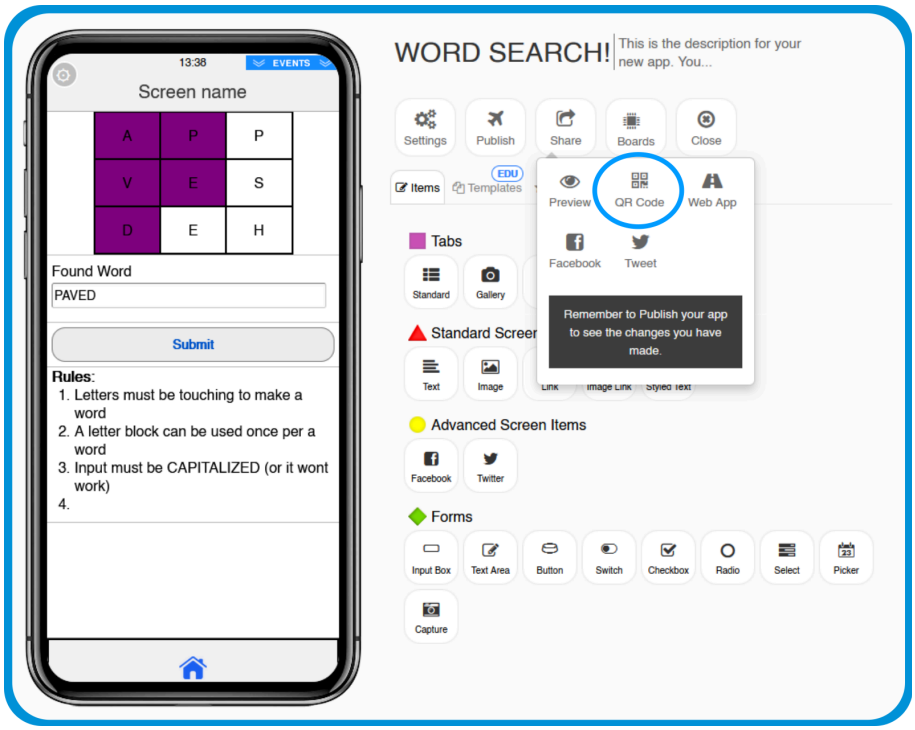
- Click **Publish**
- Click **Start**





As soon as we get a thumbs up our app is ready to view.

- Click **Share**
- Click **QR Code**
- **Scan the QR Code with your phone**



Once the code is scanned on your phone you can share this app by sending the link.

What we covered

At this point we have a fully working word search puzzle that highlights the words to show users what words they have found already! We did this by using **HTML, CSS** and finally **JavaScript** with the **jQuery** library to bring it all together

Why don't you see if you can make your own Word Search puzzles now ? Maybe even try to make a 4x4 one instead!