Grid template area
Grid areas are a way to group one or more grid cells. The grid template area is an extension of this concept where you can give names to these grid areas. Once you have the names defined, you can address these new grid area items by their names and configure them accordingly.
The property grid-template-areas is usually placed inside the body tag or any container where the grid needs to be placed, the same way that you would define the rules for the grid. The main difference is, in case of grid-template-areas the values present will be the different names.
Process

The process isn't prescriptive but these are the steps in general:

Define the grid using display property

Set the height and width of the grid

Set the grid-template-areas with the appropriate name identifiers

Add the appropriate sizes for the rows inside the grid using grid-template-rows property

Add the appropriate sizes for the columns inside the grid using grid-template-columns property

But how exactly do you use these names and where do they come from? The names that you use inside the grid template areas are the HTML tags that you have used. Or, where you need to get more specific, you designate a class name to these tags. Once the names are assigned, you define the properties for each class the same way that you define them conventionally. Let's examine an example.
Example
HTML Code:

```
<head>

    <link rel="stylesheet" href="gridta.css">

</head>



<body>

    <header> Header </header>

    <nav class="nav-bar"> Navigation </nav>

    <main> Main area </main>

    <footer> Footer </footer>

</body>
```

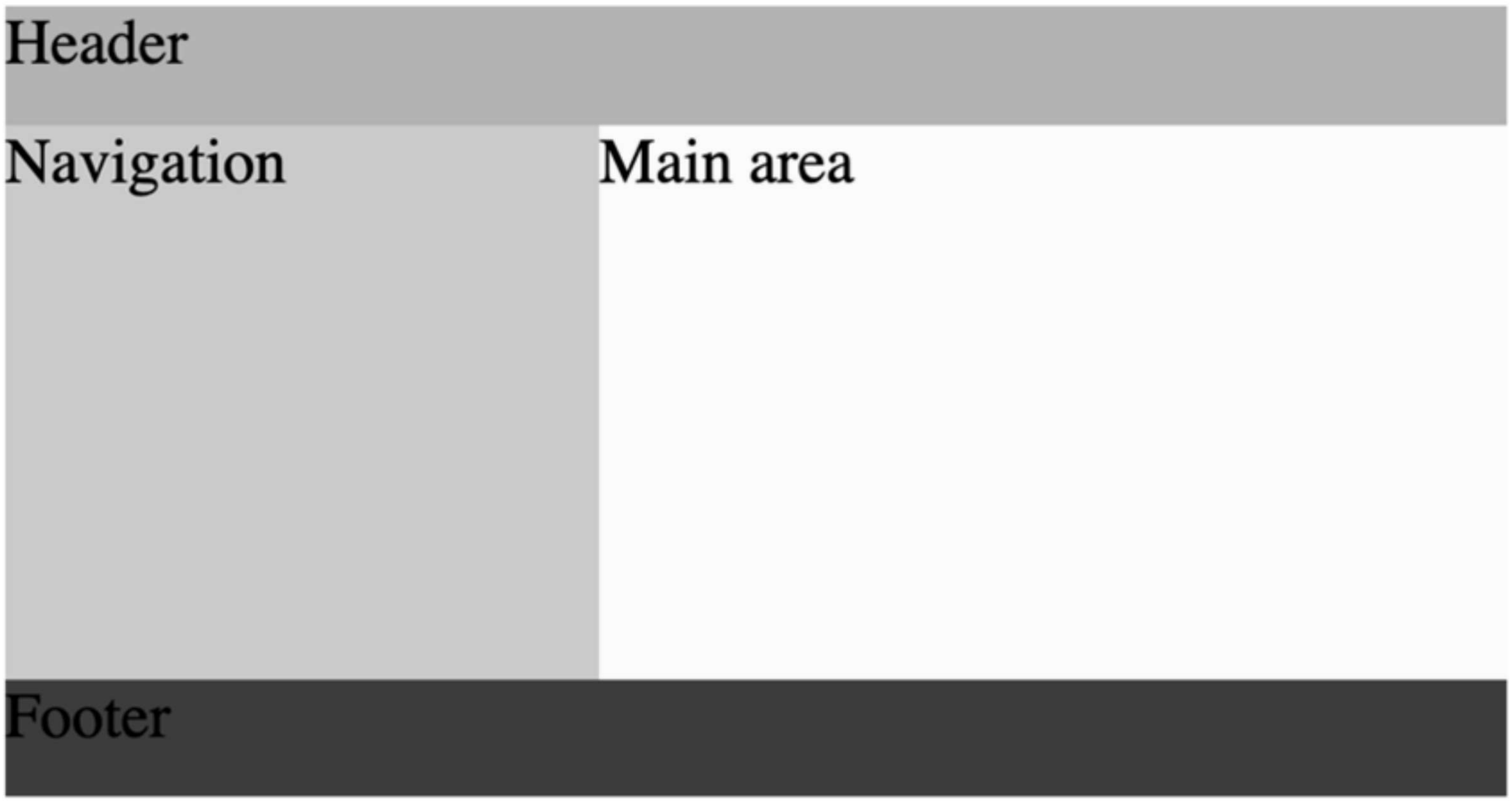
CSS Code:

```
    grid-area: footer;

    background-color: firebrick;

  }
```

Output:



Though there are five sets of rules, logically the CSS code is divided into two sections. The first is where you define the rules for the grid inside the body selector. And second is where you allocate specific rules for the different grid areas. The way these grid areas are distributed is according to how you have defined the names inside the grid-template-areas property. In the example above the relevant code is:

```
grid-template-areas: "head head"
                        "nav  main"
                        "footer  footer";
```

The ‘head’ is written twice to imply two columns and the rest of the content follows the usual convention. The number of rows will be the number of ‘pairs of quotes’ you have used which in this case is three. Namely “head head”, “nav main” and “footer footer”. Once you know the number of rows and columns, the values for those will be set by grid-template-rows and grid-template-columns. Since these are three and two respectively here, you need to add that many values. The height simply gives the overall value of the height for the grid.
Note that the number of times you wrote “header” did not have to be two. You could write more of those and if you align the rest of the grid-names correctly, the height and width of the grid-areas will be distributed proportionately.
Let’s return to the example. If you keep all other properties the same but you change the grid-template-areas as follows:

```
grid-template-areas: "head head head"
                           "nav  main main"
                           "footer  footer footer";
```

The output will remain the same as you have fixed the value of the third row to “30px”. The example is simple for the sake of clarity, but if you had used relative values, you would’ve seen an observable change in the comparable sizes of nav and main grid-areas.
Grid-areas are convenient when you have a clear schematic of what you want in a grid. It’s also easier to configure individual areas if you can address them by their names. Let’s say you are designing a resume on your website, you will be able to name the different areas such as ‘Bio’, ‘Education’, ‘Work experience’ and so on. And it’s easier to use these labels when you are defining the rules. Creating a block diagram using pen and paper before starting to work on a grid is always a good idea.