
igalia

CSS Grid Layout

Specification overview. Implementation status and roadmap.

Manuel Rego Casasnovas - rego@igalia.com
BlinkOn 2 (Zurich) - 13-14 May 2014

About Igalia & me

- Igalia is an **Open Source consultancy** founded in 2001.
- Member of **Igalia Browsers Team**.
- **Blink** and **WebKit** committer.
- Working on **CSS standards**:
  - **CSS Grid Layout**
  - CSS Regions

CSS Grid Layout

"allows authors to easily define **complex, responsive 2-dimensional layouts**"

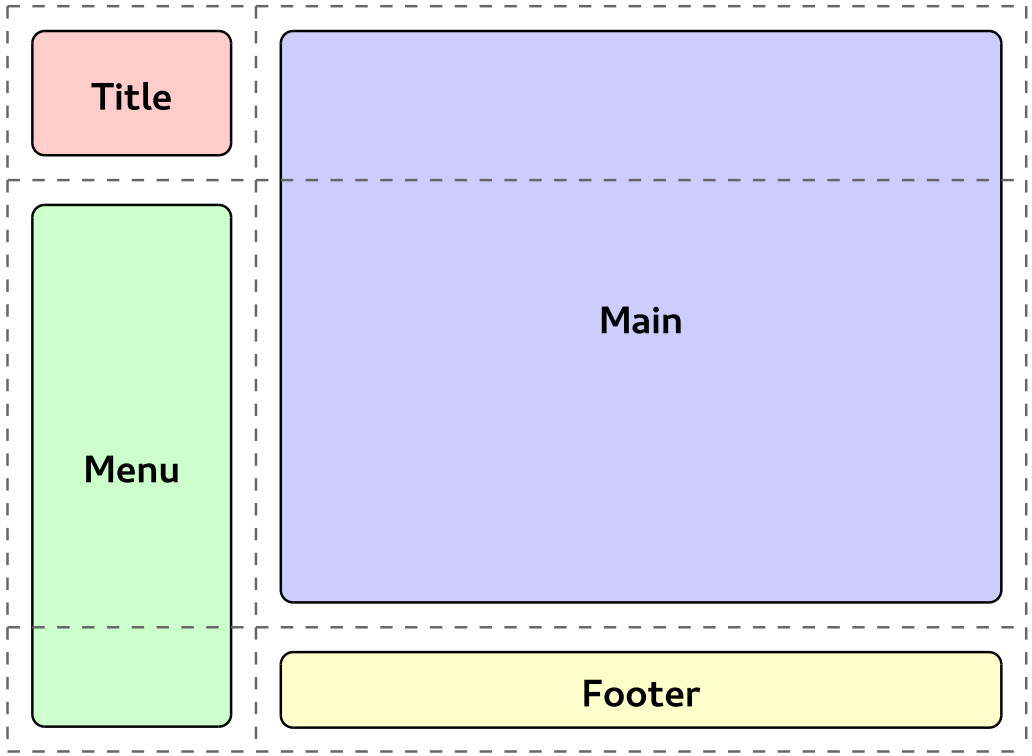
by Tab Atkins Jr. (Google)
at CSS WG Blog

source: http://www.w3.org/blog/CSS/2014/01/23/css-grid-draft-updated/

CSS Grid Layout | Introduction

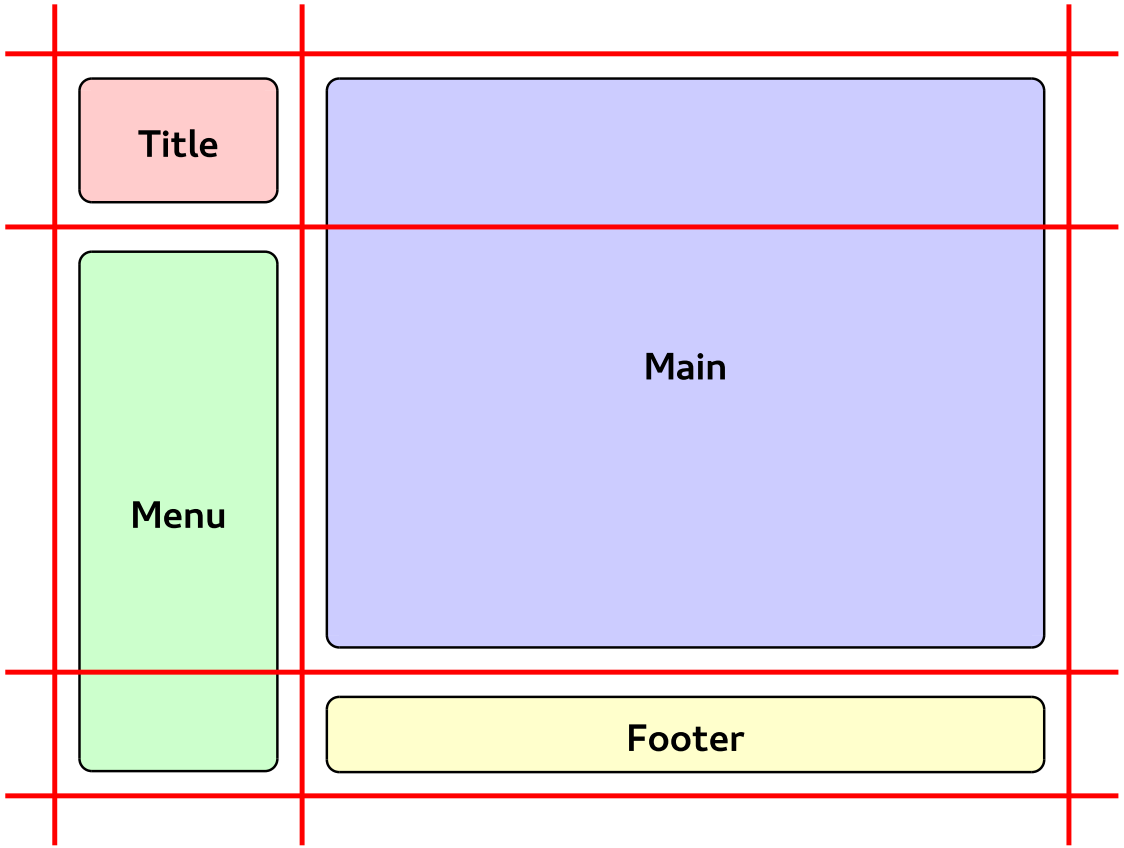
- **Grid based designs** were first done using tables and more recently floating divs.
- Those approaches have issues and a lot of complexity.
- Lots of CSS frameworks emerging to make things easier.
- CSS Grid Layout is a **powerful and flexible standard** defined by the **W3C**.

CSS Grid Layout | Introduction



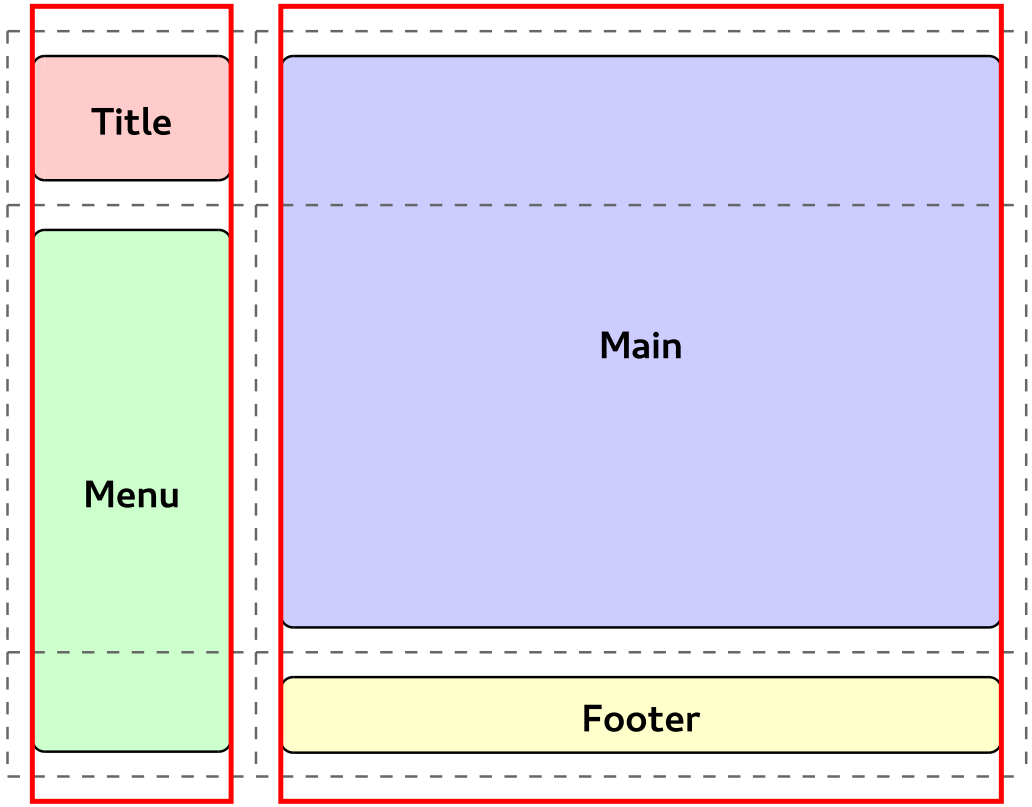
- **CSS Grid Layout** provides a mechanism to divide the available space in **rows and columns** with a set of **predictable sizing behaviors**.
- This defines a set of **grid areas** where designers can precisely place the elements of a web page.
- The Grid Layout spec can be used to intelligently **reflow elements** within a web page optimizing locations and sizes **depending on the device** where the page is rendered in.

CSS Grid Layout | Concepts



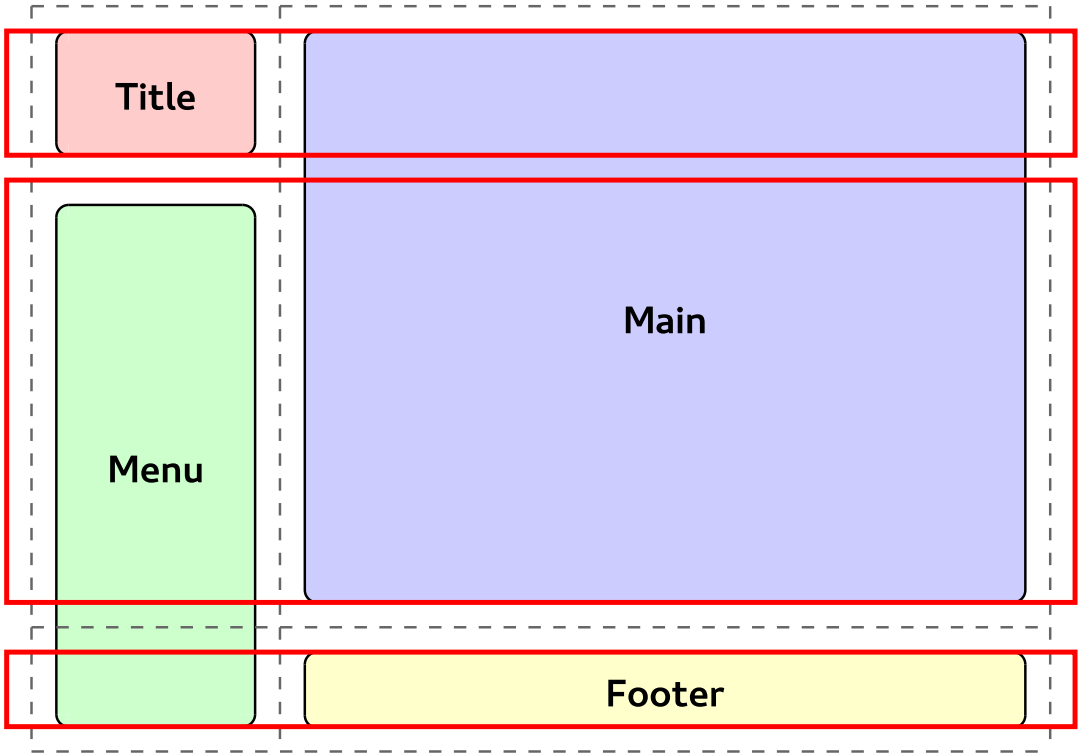
- **Grid lines** are the horizontal and vertical dividing lines of the grid.
- **Grid track** is a generic term for a grid column or grid row.
- **Grid cell** is the space between two adjacent row and two adjacent column grid lines.
- **Grid area** is the logical space used to lay out one or more grid items. It is bound by four grid lines, one on each side of the grid area.

CSS Grid Layout | Concepts



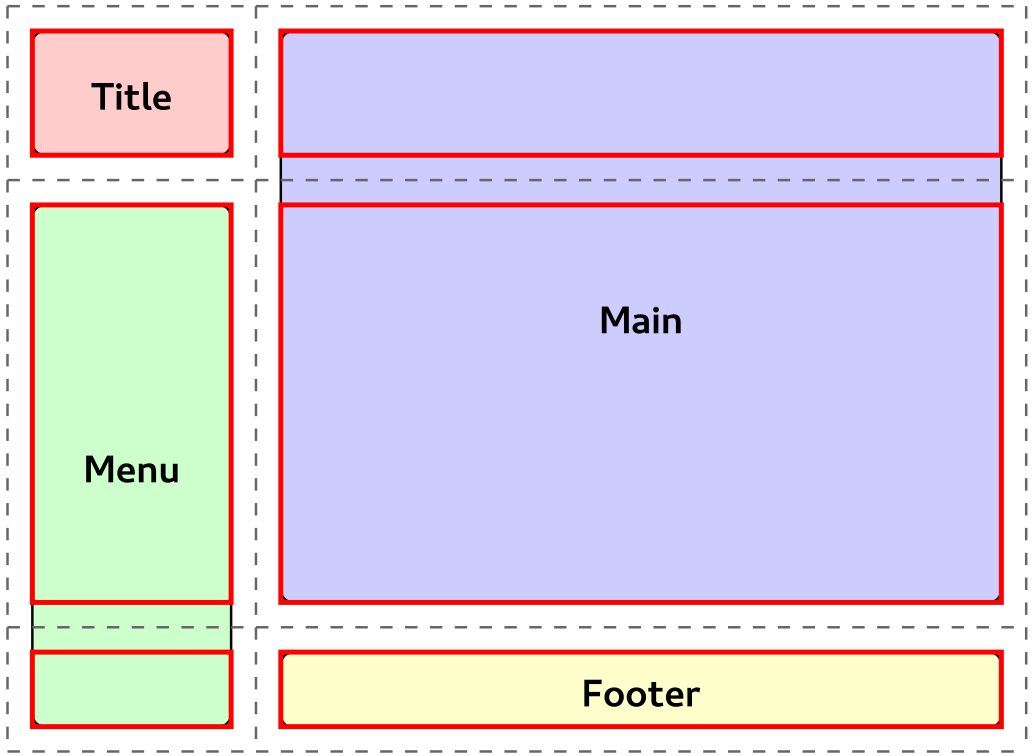
- **Grid lines** are the horizontal and vertical dividing lines of the grid.
- **Grid track** is a generic term for a grid column or grid row.
- **Grid cell** is the space between two adjacent row and two adjacent column grid lines.
- **Grid area** is the logical space used to lay out one or more grid items. It is bound by four grid lines, one on each side of the grid area.

CSS Grid Layout | Concepts



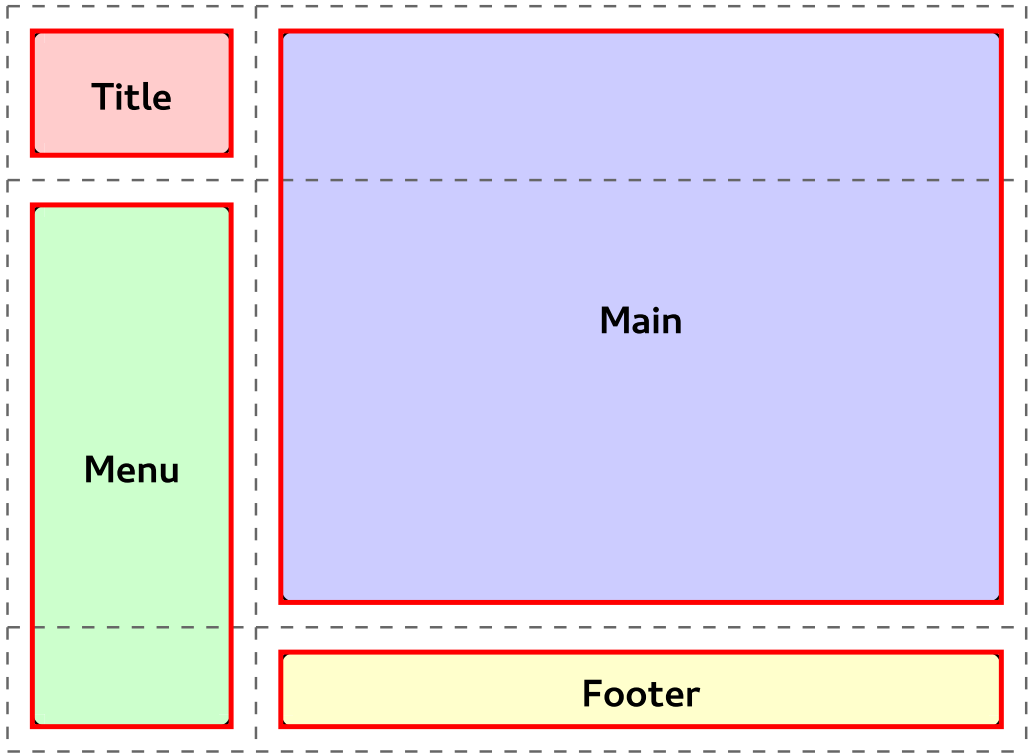
- **Grid lines** are the horizontal and vertical dividing lines of the grid.
- **Grid track** is a generic term for a grid column or grid row.
- **Grid cell** is the space between two adjacent row and two adjacent column grid lines.
- **Grid area** is the logical space used to lay out one or more grid items. It is bound by four grid lines, one on each side of the grid area.

CSS Grid Layout | Concepts



- **Grid lines** are the horizontal and vertical dividing lines of the grid.
- **Grid track** is a generic term for a grid column or grid row.
- **Grid cell** is the space between two adjacent row and two adjacent column grid lines.
- **Grid area** is the logical space used to lay out one or more grid items. It is bound by four grid lines, one on each side of the grid area.

CSS Grid Layout | Concepts



- **Grid lines** are the horizontal and vertical dividing lines of the grid.
- **Grid track** is a generic term for a grid column or grid row.
- **Grid cell** is the space between two adjacent row and two adjacent column grid lines.
- **Grid area** is the logical space used to lay out one or more grid items. It is bound by four grid lines, one on each side of the grid area.

CSS Grid Layout | Syntax

CSS

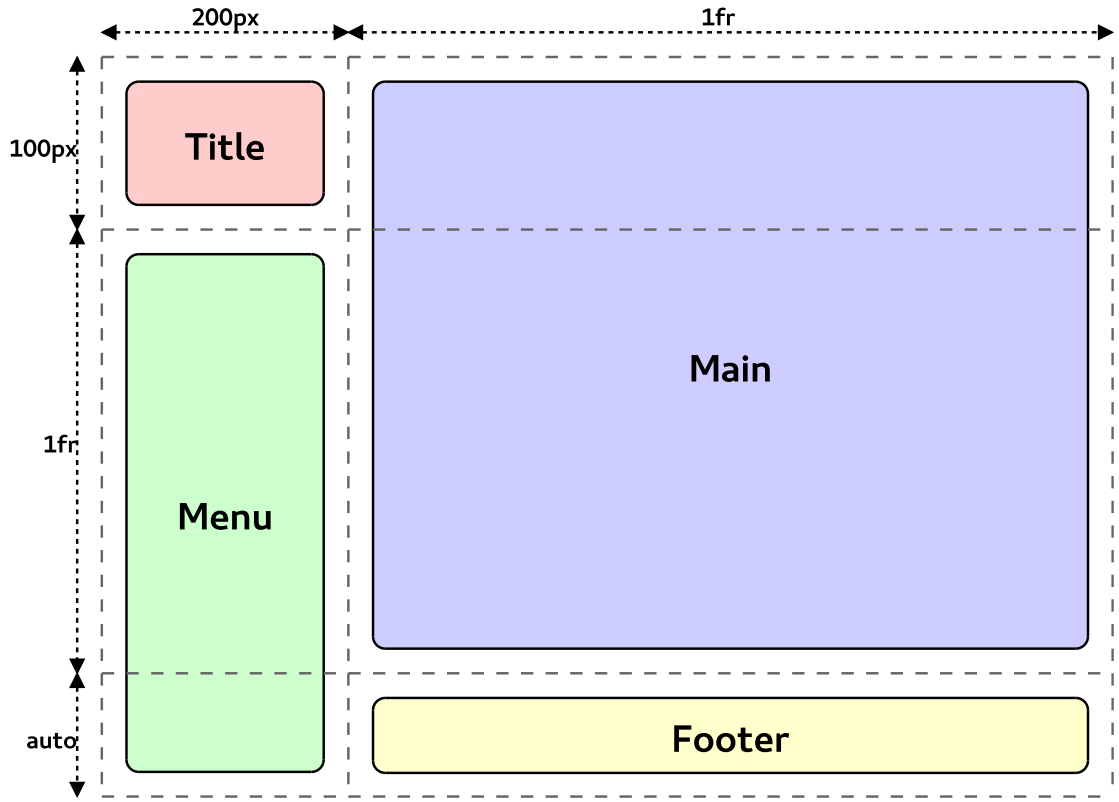
```css
.grid {
    display: grid;
    grid-template-columns: 200px 1fr;
    grid-template-rows: 100px 1fr auto;
}

.title  { grid-column: 1; grid-row: 1;          }
.menu   { grid-column: 1; grid-row: 2 / span 2; }
.main   { grid-column: 2; grid-row: 1 / span 2; }
.footer { grid-column: 2; grid-row: 3;          }
```

- `display: grid`: Defines a grid container.
- `grid-template-columns` and `grid-template-rows`: Specify the track breadths.
- `grid-column` and `grid-row`: Determine a grid item's size and location within the grid.

HTML

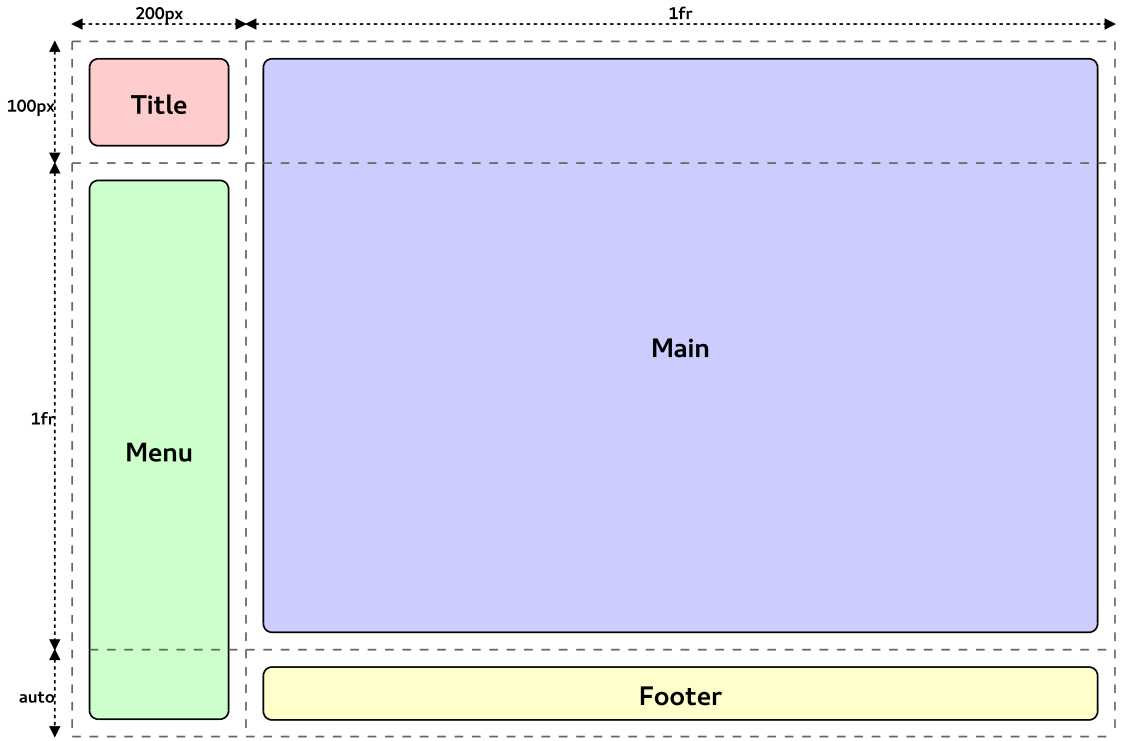
```html
<div class="grid">
    <div class="title">Title</div>
    <div class="menu">Menu</div>
    <div class="main">Main</div>
    <div class="footer">Footer</div>
</div>
```

CSS Grid Layout | Track Breadths



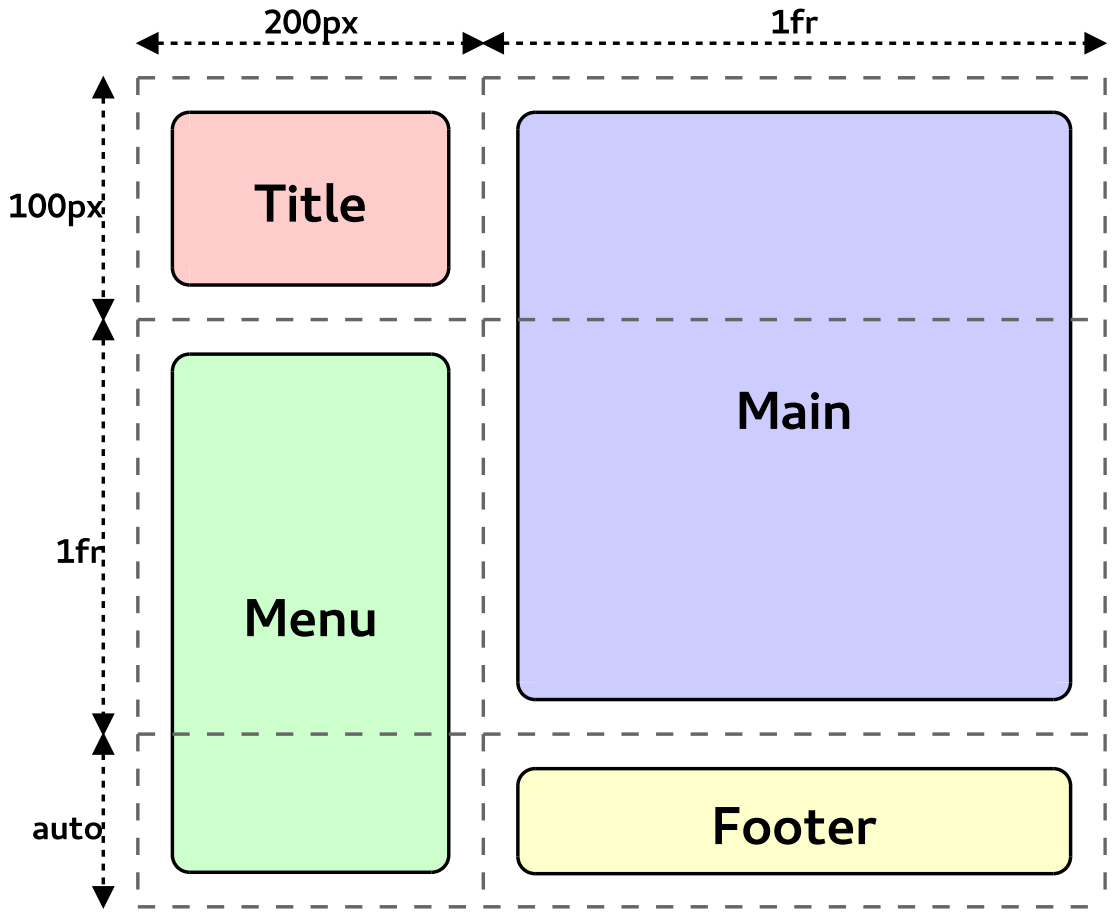
- **Options**:
- length
- percentage
- flex (`fr` - free space - unit)
- `max-content`
- `min-content`
- `minmax(min, max)`
- `auto`

CSS Grid Layout | Flexible Track Breadths



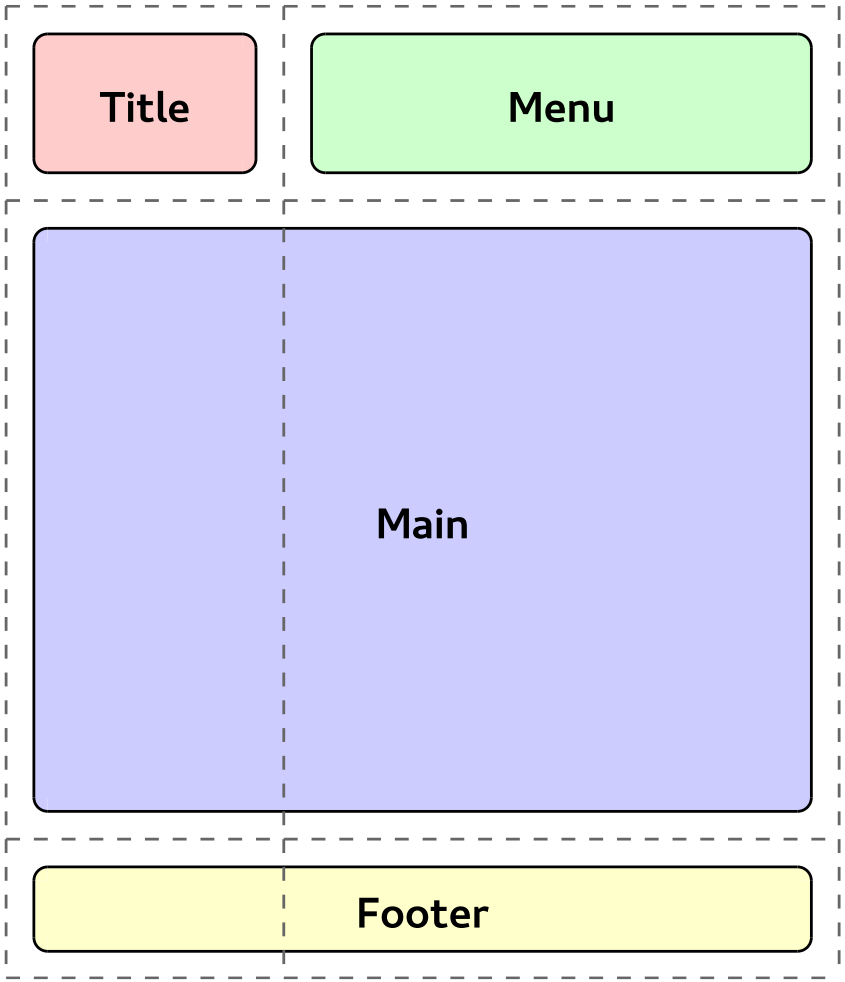
- Flexible track breadths will **grow or shrink** and automatically adapt themselves to viewport size.

CSS Grid Layout | Flexible Track Breadths



- This allows to create very **customized grids**, defining flexible tracks with minimum and maximum breadths depending on their contents or the available free space left by the rest of tracks and viewport size.

CSS Grid Layout | Responsive Layout



- CSS Grid Layout can be used combined with other CSS features to create **responsive designs**.
- For example previous case can be adapted to portrait mode if the screen orientation changes using Media Queries.

CSS Grid Layout | Responsive Layout

CSS

```
@media (orientation: portrait) {
    .grid {
        grid-template-columns: auto 1fr;
        grid-template-rows: auto 1fr auto;
    }

    .title  { grid-column: 1; grid-row: 1;          }
    .menu   { grid-column: 2; grid-row: 1;          }
    .main   { grid-column: 1 / span 2; grid-row: 2; }
    .footer { grid-column: 1 / span 2; grid-row: 3; }
}
```

- Change `grid-template-columns` and `grid-template-rows`.
- Change position inside the grid of the different items.

CSS Grid Layout | Examples

- **Enable experimental Web Platform features**: chrome://flags/#enable-experimental-web-platform-features
- **GitHub repository** with different examples: **http://igalia.github.io/css-grid-layout/**
  - **Responsive grid**: http://igalia.github.io/css-grid-layout/responsive-grid.html
  - **Gallery**: http://igalia.github.io/css-grid-layout/gallery.html
- **Video**: https://www.youtube.com/watch?v=nsWtMN54bEI

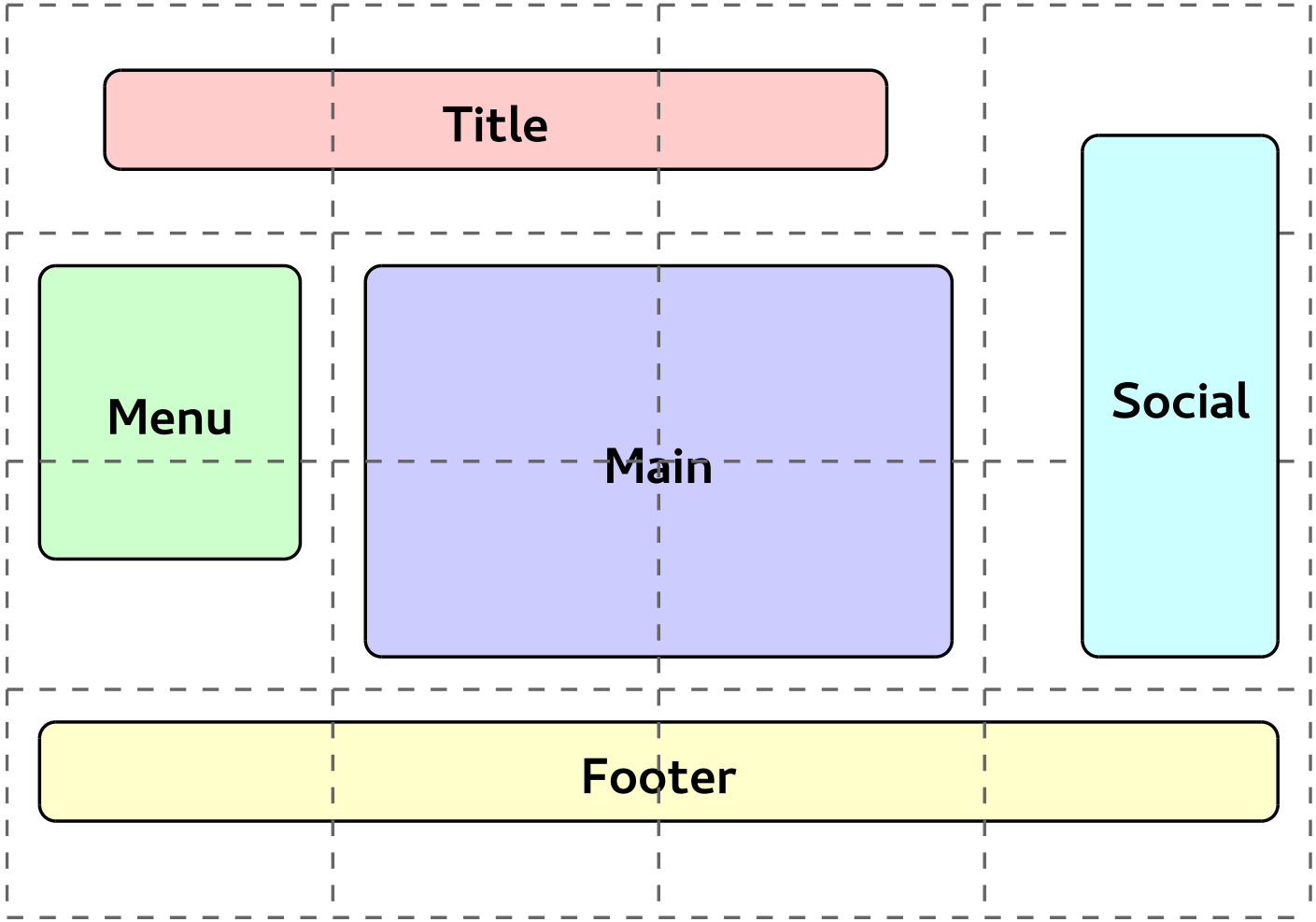
CSS Grid Layout | Areas & Alignment

CSS

```
.grid {
    display: grid;
    grid-template-areas: "title  title  title  social"
                         "menu   main   main   social"
                         "menu   main   main   social"
                         "footer footer footer footer";
}
.title  { grid-area: title; align-self: center; justify-self: center; }
.menu   { grid-area: menu;  align-self: start;                        }
.main   { grid-area: main;                                            }
.social { grid-area: social; align-self: end;   justify-self: right;  }
.footer { grid-area: footer; align-self: start;                       }
```

- `grid-template-areas` specifies named grid areas that can be referenced to position grid items.
- Follows CSS Box Alignment spec for alignment features.

CSS Grid Layout | Areas & Alignment



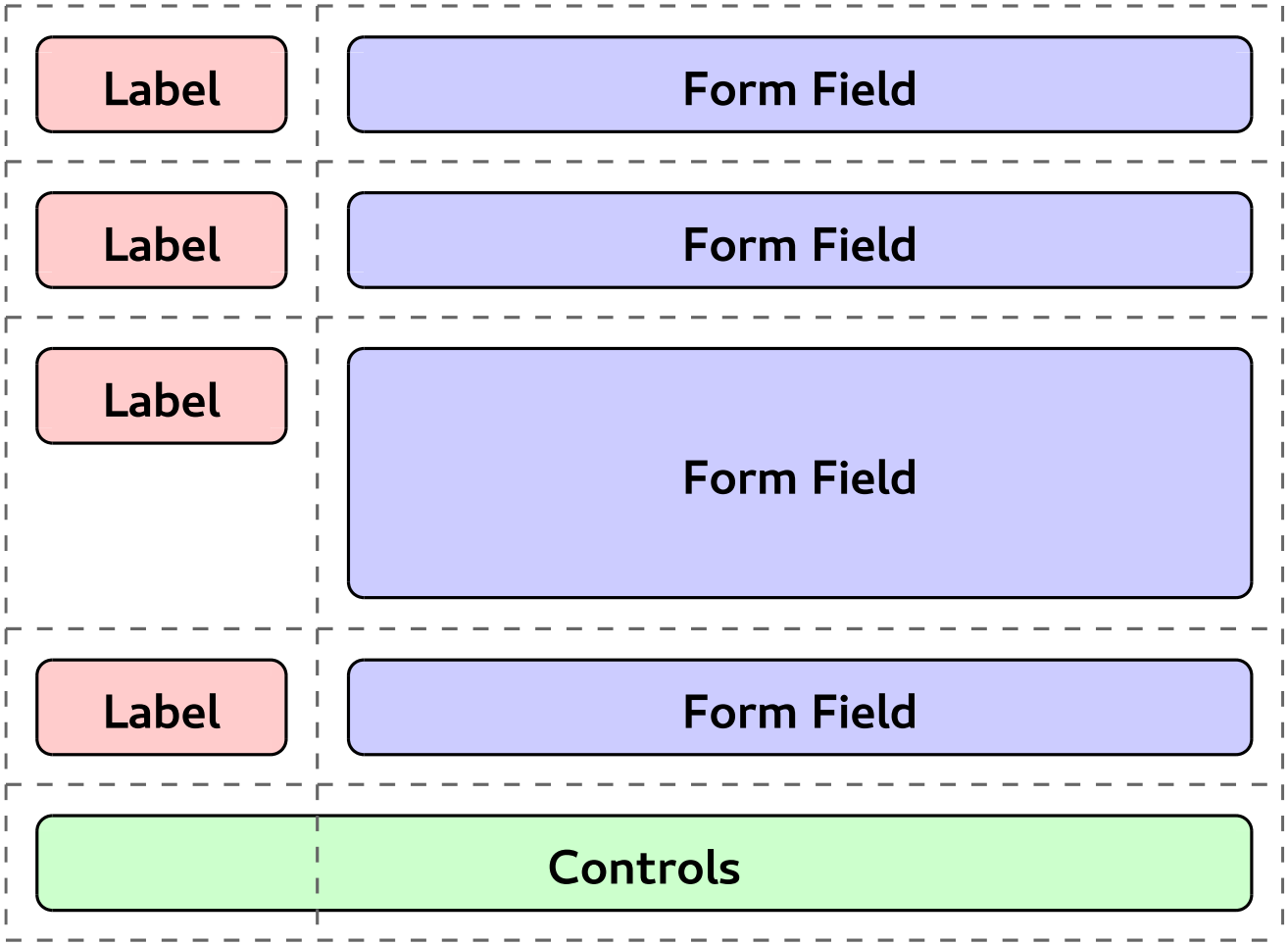
CSS Grid Layout | Auto-placement

CSS

```
form {
    display: grid;
    grid-auto-flow: row;
}
label           { grid-column: 1;          }
input, textarea { grid-column: 2;          }
.controls       { grid-column: 1 / span 2; }
```

- Grid items can be positioned outside of the grid bounds. This causes the grid container to generate implicit grid tracks, forming the **implicit grid**.
- `grid-auto-flow` controls the direction in which the grid items are automatically placed.

CSS Grid Layout | Auto-placement



CSS Grid Layout | Current status

- Spec (W3C Working Draft, 23 January 2014): http://www.w3.org/TR/css-grid-1/.
- Main browsers:
  - Old version already **shipped in IE/Trident**.
  - Work in progress in Chromium/Blink (Google and Igalia) and Safari/WebKit (Igalia).
  - Mozilla has started the development in Firefox/Gecko early this year.

CSS Grid Layout | Implementation Status

- Define grids using all different **properties** in the spec.
- Support for **named grid lines** and **named grid areas**.
- All **placement** options are supported too (both explicit and implicit grid).
- **Track sizing** and **auto-placement algorithms** are already implemented.

CSS Grid Layout | Roadmap

- Subgrids (still under discussion inside the CSS WG).
- Alignment.
- Fragmentation.
- Test coverage.
- Performance optimizations.
- Support for different writing modes.
- Selection.

Acknowledgements

Bloomberg

- **Bloomberg** is sponsoring our work in CSS Grid Layout.

Thank You!

igalia.com/browsers

g+	google.com/+ManuelRegoCasasnovas
twitter	@regocasasnovas
www	people.igalia.com/mrego/
github	github.com/mrego