# Adapting Desktop Web Sites for Mobile Devices

## Contents

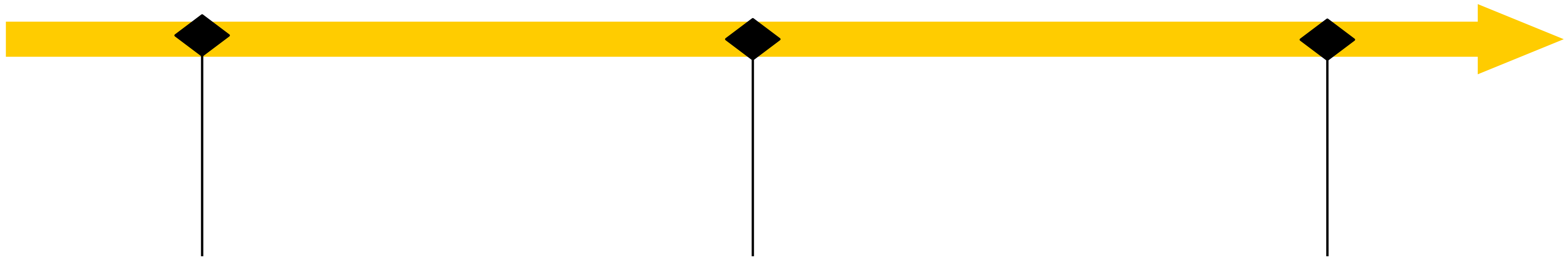
1. Understanding Content Adaptation Approaches

2. Controlling Viewport Layout

3. Using CSS Media Queries

4. Designing Forms

## 1. Content adaptation strategies



**“Do Nothing”**

Existing desktop content is served directly to mobile devices without any layout changes or optimizations (except maybe Viewport)

**Multi-Serve**

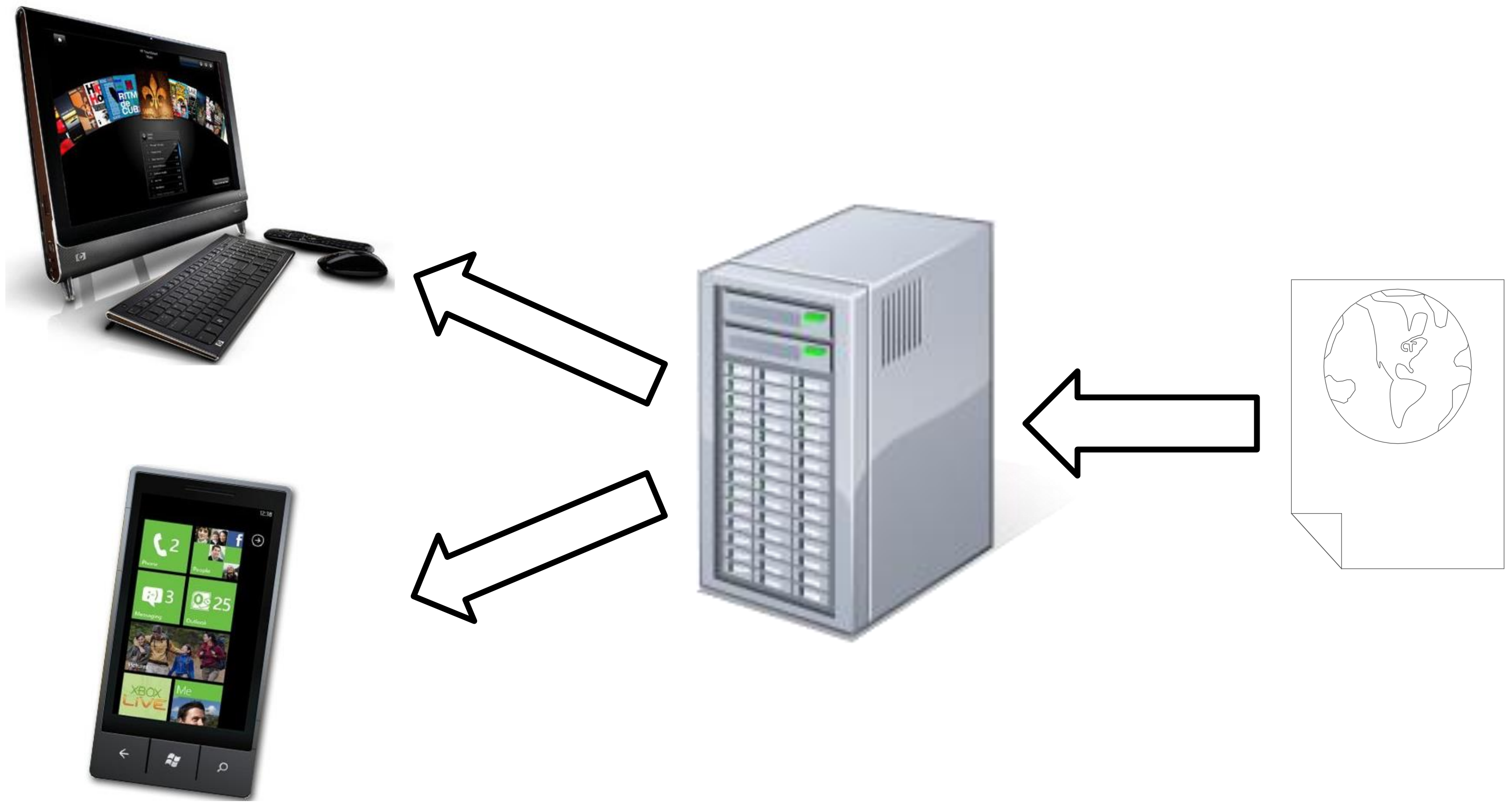
The same page content is served to both desktop and mobile devices, but with styling and script appropriate for the target form factor

**Mobile-Specific**

Mobile-specific content is created and served to mobile devices while desktop devices receive desktop-appropriate content

## The “Do nothing” content adaptation approach

- Desktop-class content is served to mobile device un-altered



## The “Do nothing” content adaptation approach

**Advantages**

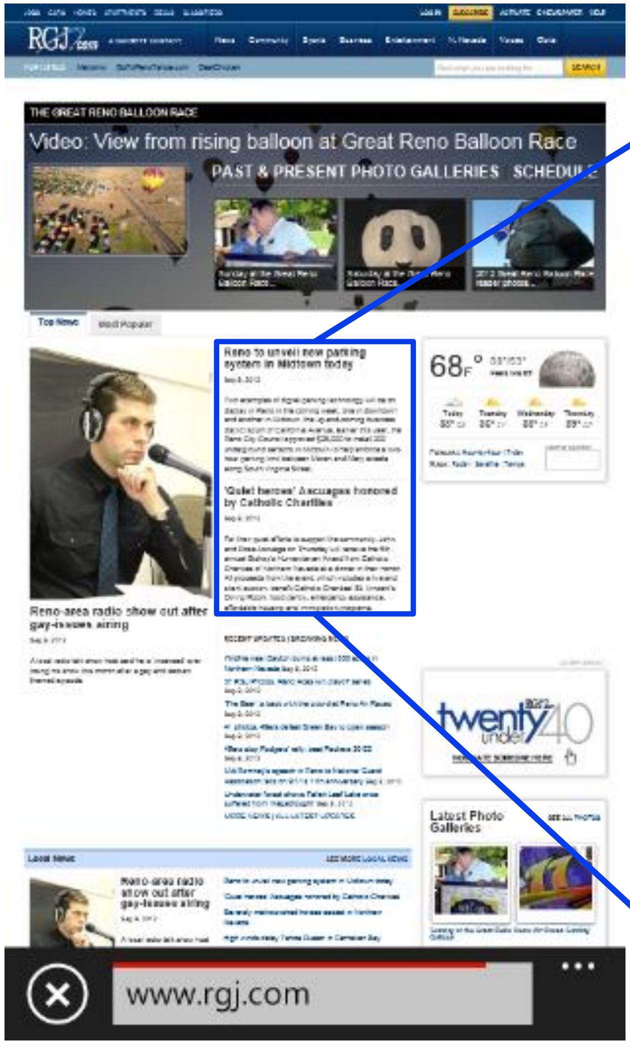
- Less work to develop
- No need to maintain separate copies of content or styles

**Disadvantages**

- Not result in an optimal experience for users
- Complex content can cause poor performance

- **OK FOR**: lightweight pages that have flexible, flowing content; pages that are arranged in traditional vertical columns
- **NOT GOOD FOR**: pages with complex layouts; pages that have large content items (like big images) or rely on plug-ins like Flash

# The “Do nothing” approach: examples


THE GREAT RENO BALLOON RACE

Video: View from rising balloon at Great Reno Balloon Race

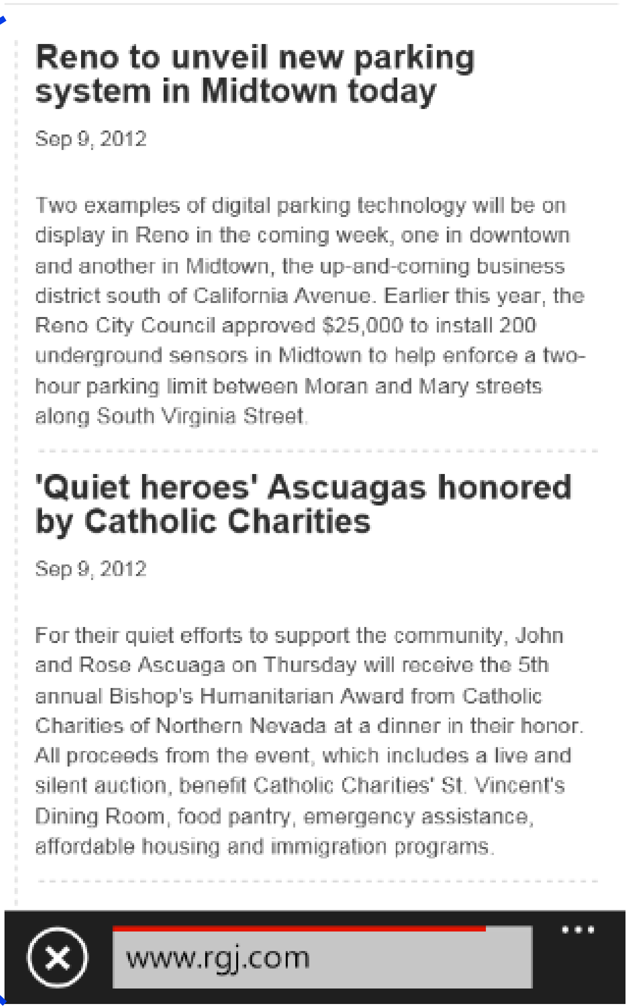
PAST & PRESENT PHOTO GALLERIES SCHEDULE

Reno-area radio show out after gay-issues airing

68°F

Latest Photo Galleries

www.rgj.com


**Reno to unveil new parking system in Midtown today**

Sep 9, 2012

Two examples of digital parking technology will be on display in Reno in the coming week, one in downtown and another in Midtown, the up-and-coming business district south of California Avenue. Earlier this year, the Reno City Council approved $25,000 to install 200 underground sensors in Midtown to help enforce a two-hour parking limit between Moran and Mary streets along South Virginia Street.

**'Quiet heroes' Ascuagas honored by Catholic Charities**

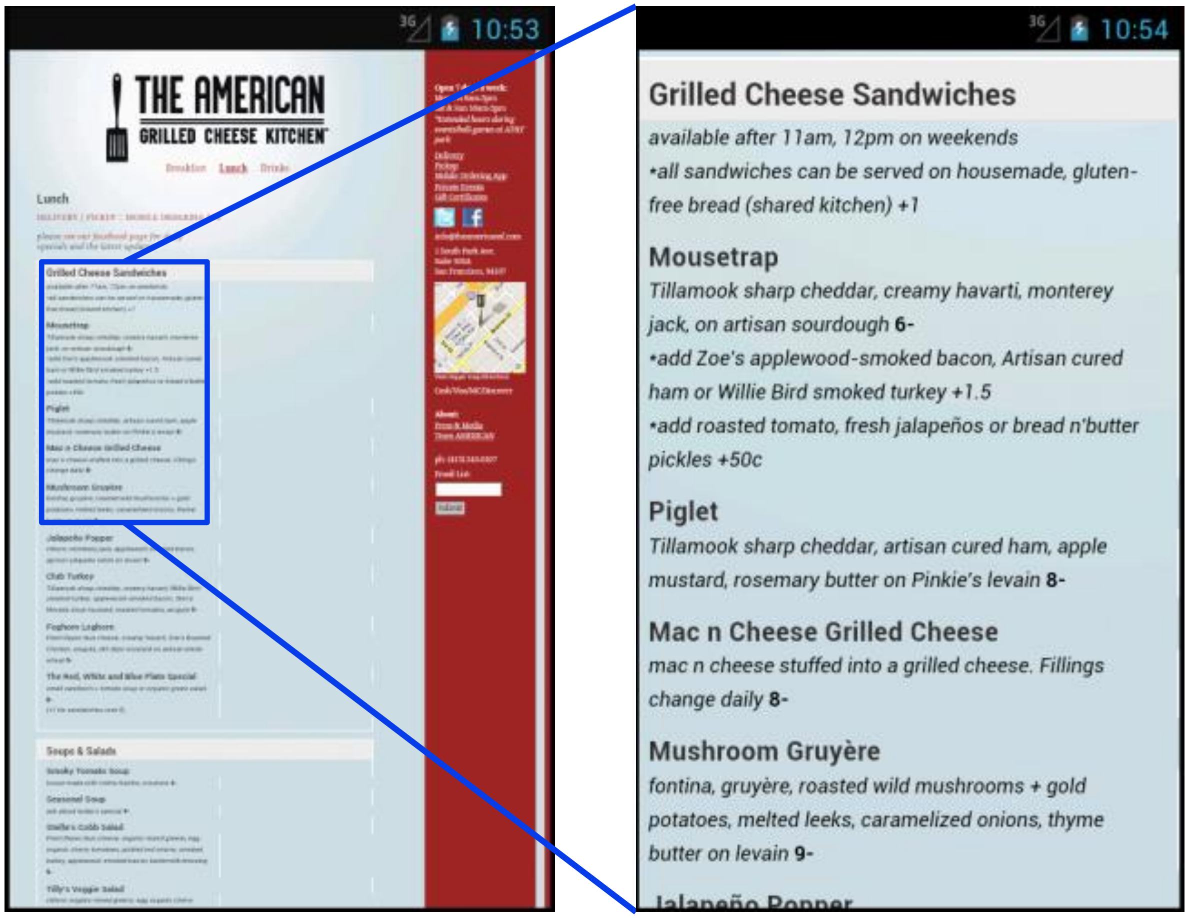
Sep 9, 2012

For their quiet efforts to support the community, John and Rose Ascuaga on Thursday will receive the 5th annual Bishop's Humanitarian Award from Catholic Charities of Northern Nevada at a dinner in their honor. All proceeds from the event, which includes a live and silent auction, benefit Catholic Charities' St. Vincent's Dining Room, food pantry, emergency assistance, affordable housing and immigration programs.

www.rgj.com

If double-click, the content zooms in and becomes pretty readable

# The “Do nothing” approach: examples



Grilled Cheese Sandwiches

*available after 11am, 12pm on weekends*

*•all sandwiches can be served on housemade, gluten-free bread (shared kitchen) +1*

Mousetrap

*Tillamook sharp cheddar, creamy havarti, monterey jack, on artisan sourdough* **6-**

*•add Zoe's applewood-smoked bacon, Artisan cured ham or Willie Bird smoked turkey +1.5*

*•add roasted tomato, fresh jalapeños or bread n'butter pickles +50c*

Piglet

*Tillamook sharp cheddar, artisan cured ham, apple mustard, rosemary butter on Pinkie's levain* **8-**

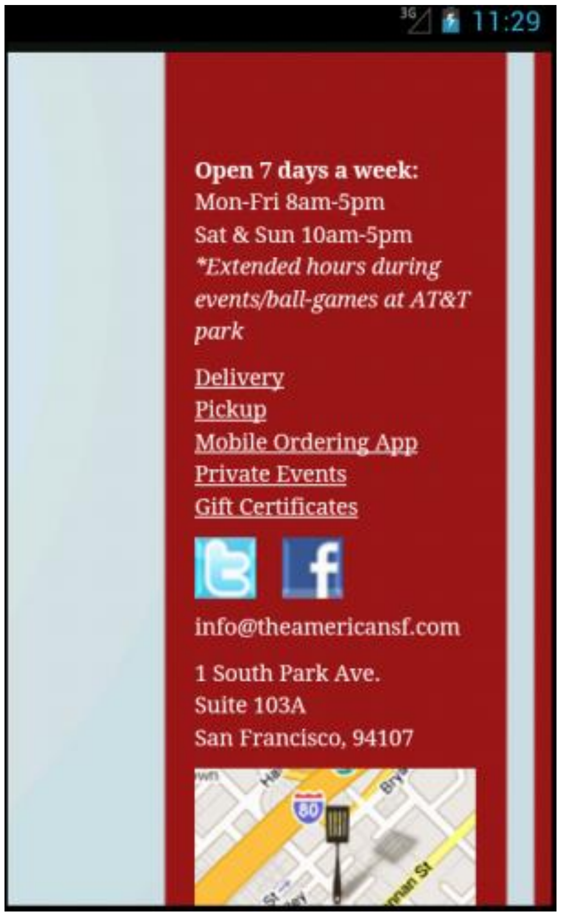
Mac n Cheese Grilled Cheese

*mac n cheese stuffed into a grilled cheese. Fillings change daily* **8-**

Mushroom Gruyère

*fontina, gruyère, roasted wild mushrooms + gold potatoes, melted leeks, caramelized onions, thyme butter on levain* **9-**

Jalapeño Popper



**Open 7 days a week:**
Mon-Fri 8am-5pm
Sat & Sun 10am-5pm
**Extended hours during events/ball-games at AT&T park*

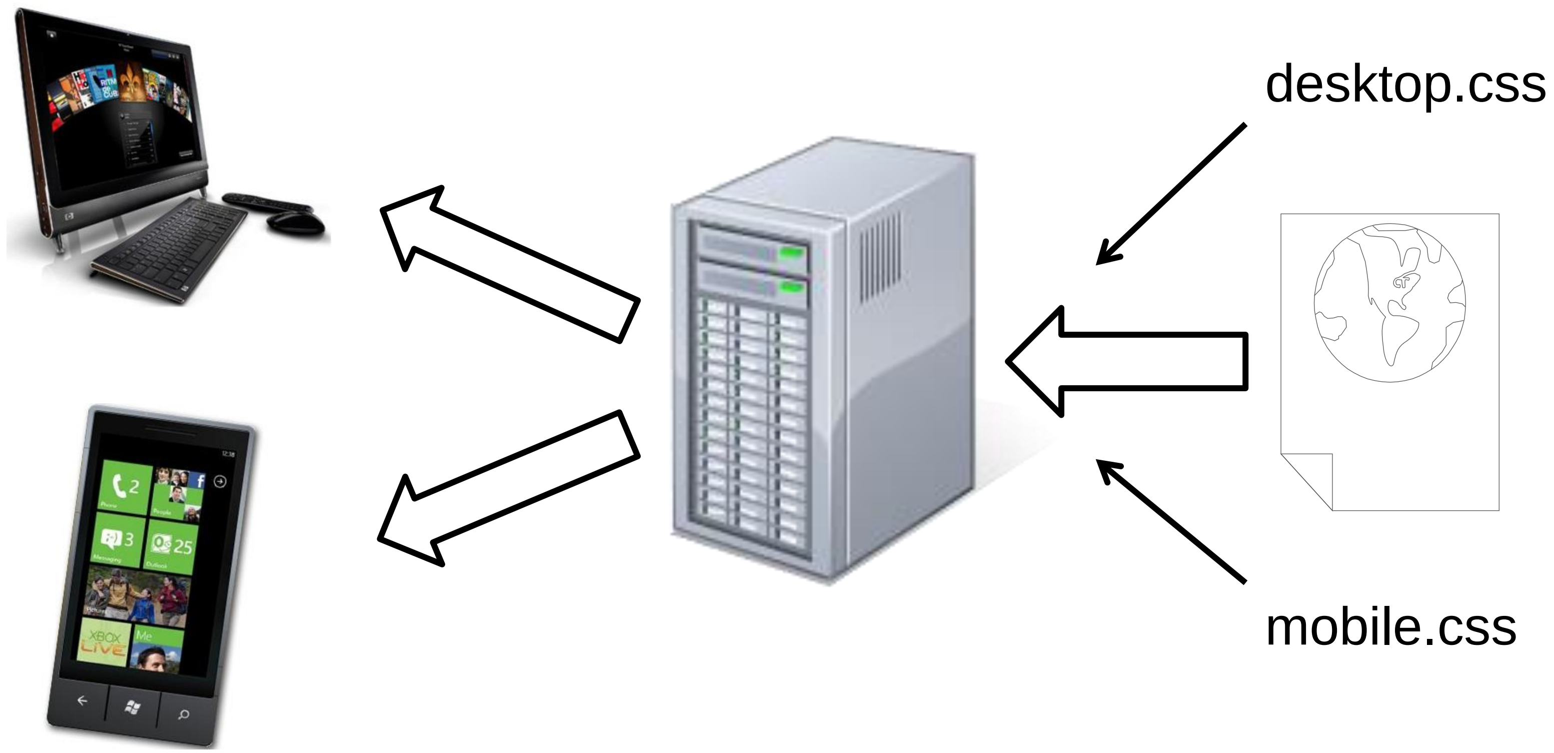
Delivery
Pickup
Mobile Ordering App
Private Events
Gift Certificates

info@theamericansf.com

1 South Park Ave.
Suite 103A
San Francisco, 94107

## The Multi-serve content adaptation approach

- Deliver the same page content to mobile and desktop devices
- However, CSS and JavaScript can be tailored to the form factor
  - Based upon what device is making the request, insert either a desktop styling or mobile styling into the response



## The Multi-serve content adaptation approach

**Advantages**

- Using the same content reduces development work
- Good experiences for multiple form factors

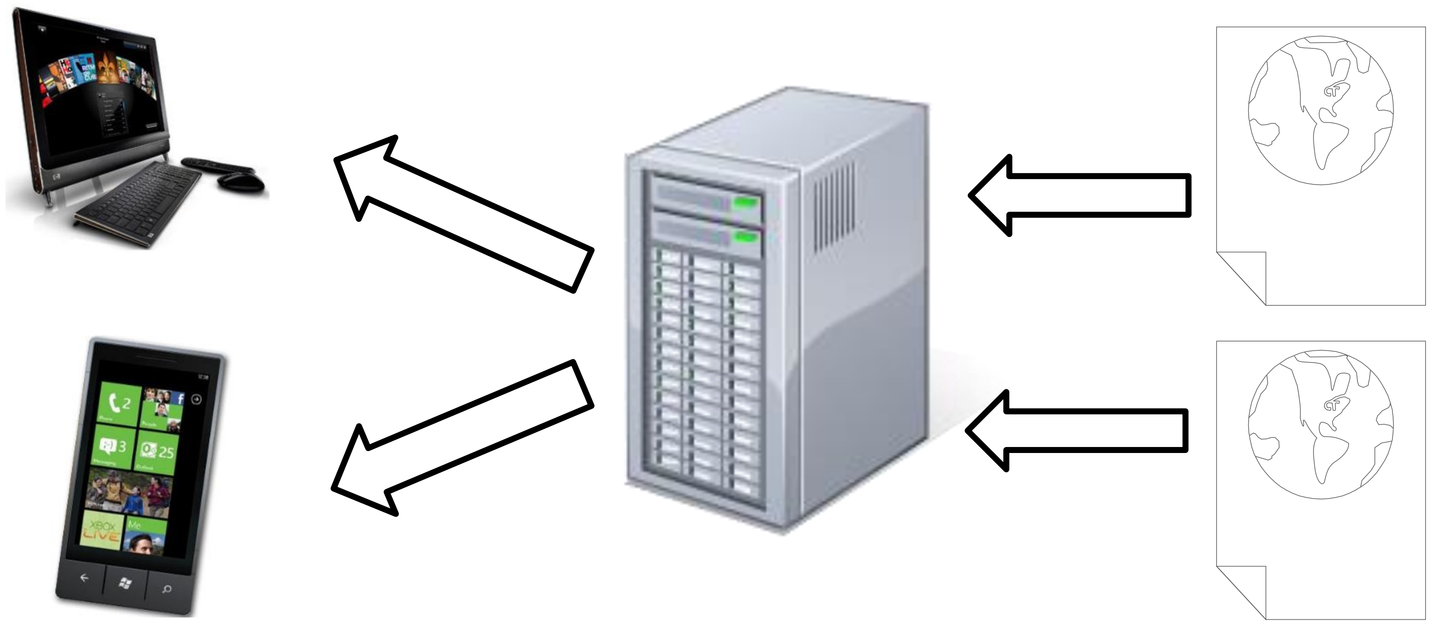
**Disadvantages**

- Can be difficult to convert existing content to use this model, based upon how complex the page is
- Careless use of styles can defeat the benefits of this approach

- **OK FOR**: Pages that are mostly semantic markup and use style sheets/scripts to define appearance
- **NOT GOOD FOR**: Complex pages with embedded images or video; pages whose content differs greatly between desktop and mobile

## The Mobile-specific content adaptation approach

- Mobile-specific content is built and delivered to mobile
- Involves some kind of server-side detection & redirection



# The Multi-serve content adaptation approach

**Advantages**

- Each form factor gets the best possible experience
- Pages can be developed independently from each other

**Disadvantages**

- Increases development work, since pages are built more than once
- Usually relies on server-side detection, which can be unreliable

- **OK FOR**: Pages that are complex; where experience differs greatly from one form factor to another; sites where mobile & desktop are independently maintained
- **NOT GOOD FOR**: Simpler pages that can be more easily addressed with one of the other content adaptation approaches

## Contents

1. Understanding Content Adaptation Approaches

2. Controlling Viewport Layout

3. Using CSS Media Queries

4. Designing Forms

## Designating mobile-ready pages with the Viewport

- Mobile browsers often try to optimize desktop Web pages to work well
- The following identifiers *turn off* any optimizations that the mobile browser might be trying to do

```
<META name=“HandheldFriendly” content=“true” />
<META name=“MobileOptimized” content=“320” />
<META name=“Viewport” content=“width=device-width” />
```

## Designating mobile-ready pages with the Viewport

- <META name=“HandheldFriendly” content=“true” />
  - "Hey! This is a mobile web page. I don't need to do any special work. The designer has designed it to work well on a mobile browser.”

## Designating mobile-ready pages with the Viewport

- <META name="MobileOptimized" content="320" />
  - More advanced than the HandheldFriendly tag
  - Specify the screen width in pixels that the page was designed for
  - "Hey! This page was designed to be well rendered and laid out in a window that's 320pixels wide."
  - Content = "0" : laid out on whatever the width of the screen is

## Designating mobile-ready pages with the Viewport

- <META name=“Viewport” content=“width=device-width” />
    - Says what the optimal width (or height) is
    - "Whatever the width of the screen is, that's what I'm laid out for."
        - "The width of this page should be whatever the device width happens to be"
    - Most recent  and de facto standard
        - While invented by Apple, most mobile browsers support

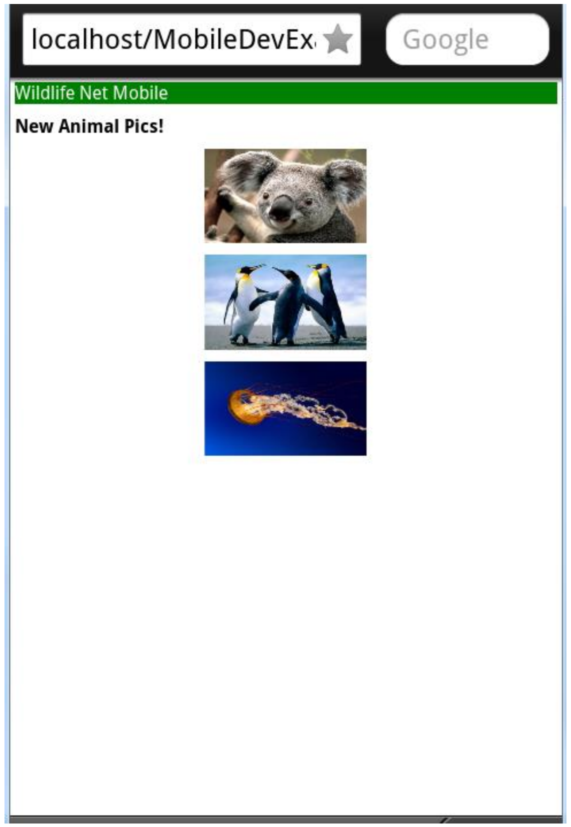
## The Viewport meta tag

- Controls how the browser displays the page
  - Its presence also disables the browser’s mobile optimizations for pages
- Has several properties that determine how the browser responds to user interaction

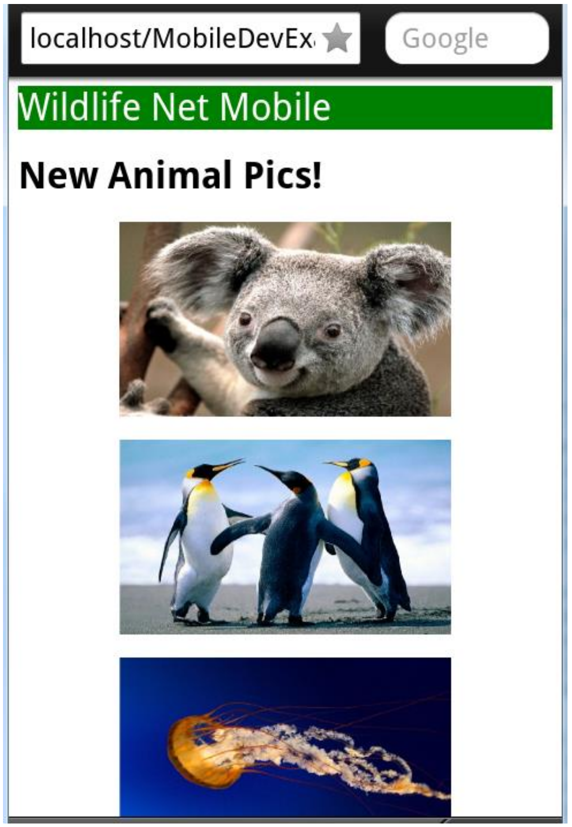
## The Viewport meta tag: properties

| Viewport Property | Description |
|---|---|
| **width** | Sets the width of the viewport. Can be either an integer value or “device-width”. Default values are: Windows Phone 7 – 1024, iPhone – 980, Android – 800, and Opera Mobile – 800 pixels |
| **height** | Sets the height of the viewport. Can be either an integer value or “device-height” |
| **initial-scale** | Sets the initial scale value of the viewport. Default to 1.0 |
| **maximum -scale** | Determines the maximum scale value of the viewport. Range is from 0.1 to 10.0 |
| **minimum-scale** | Determines the minimum scale value of the viewport . Range is from 0.1 to 10.0 |
| **user-scalable** | Boolean  that determines whether the user can scale the viewport. Default is true |

## Example of the Viewport tag

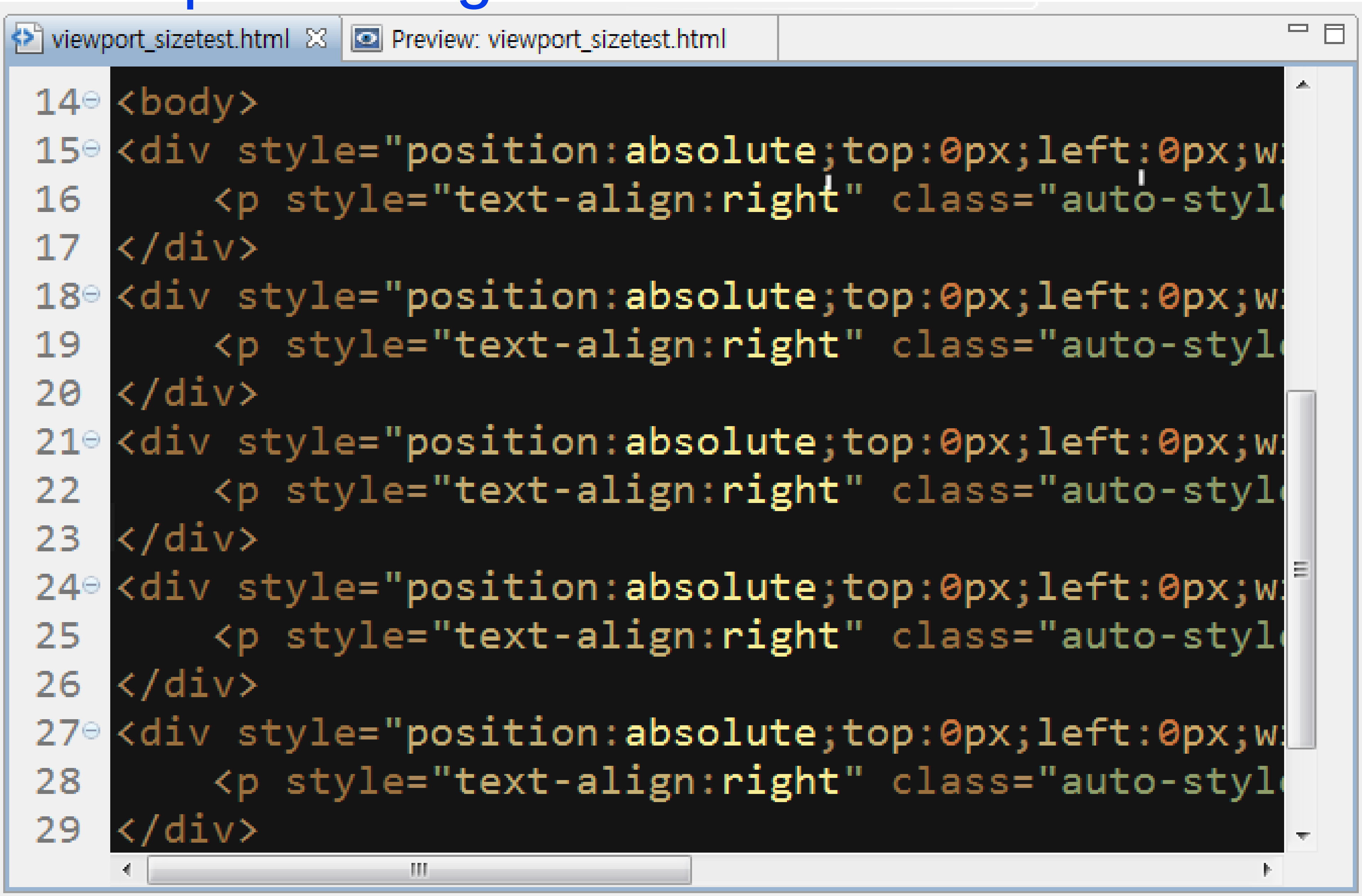


No viewport tag – the page is displayed zoomed out because the browser assumes a width of 800-1000 pixels

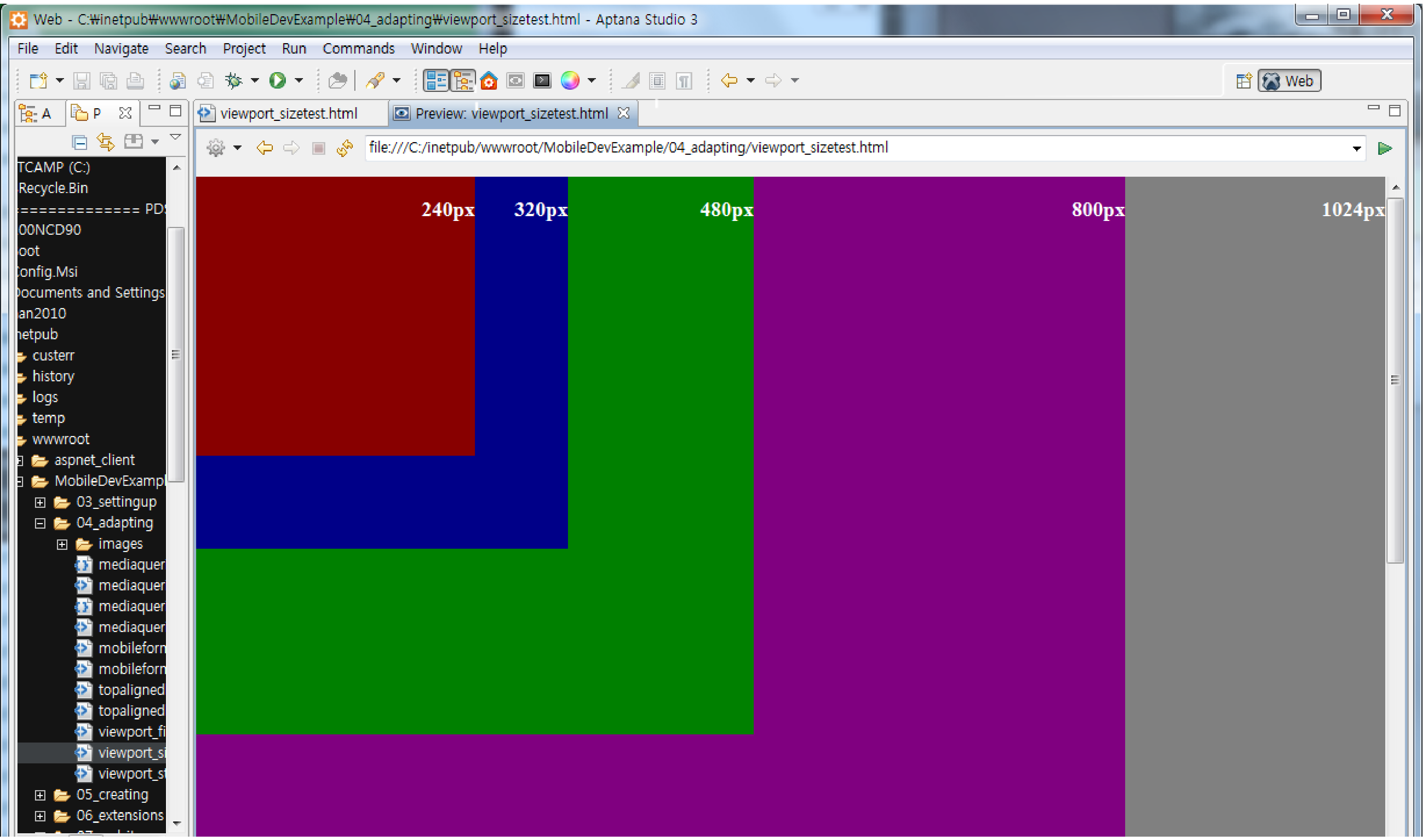


With a viewport tag – the page is correctly a size for the device using “device-width”, or for a specific pixel width like “320” or “480”

# Viewport sizing test

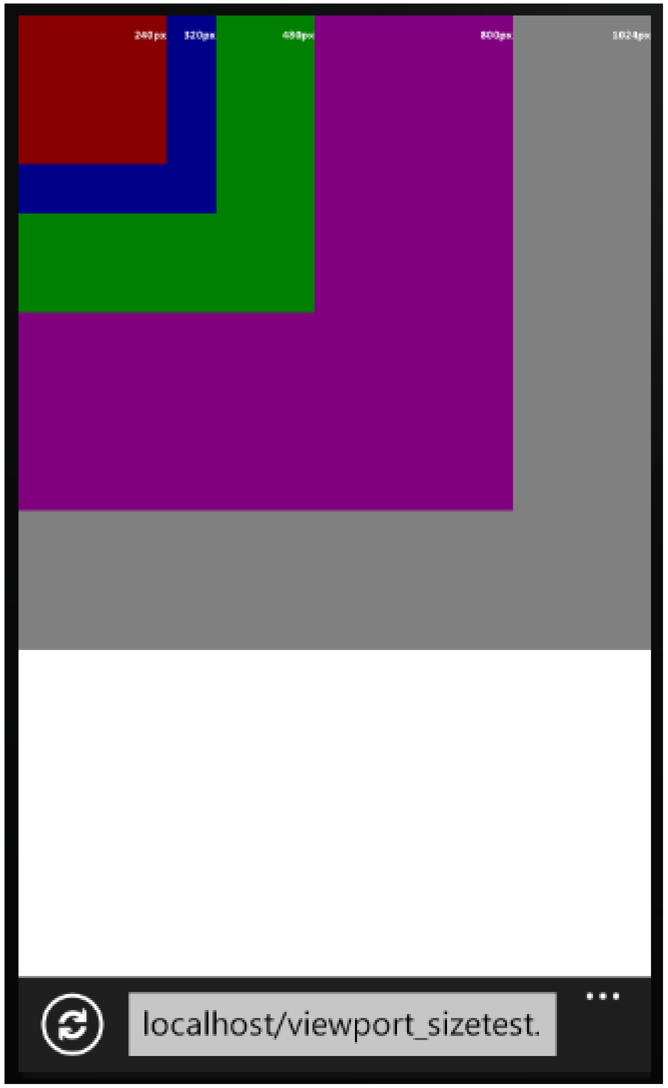


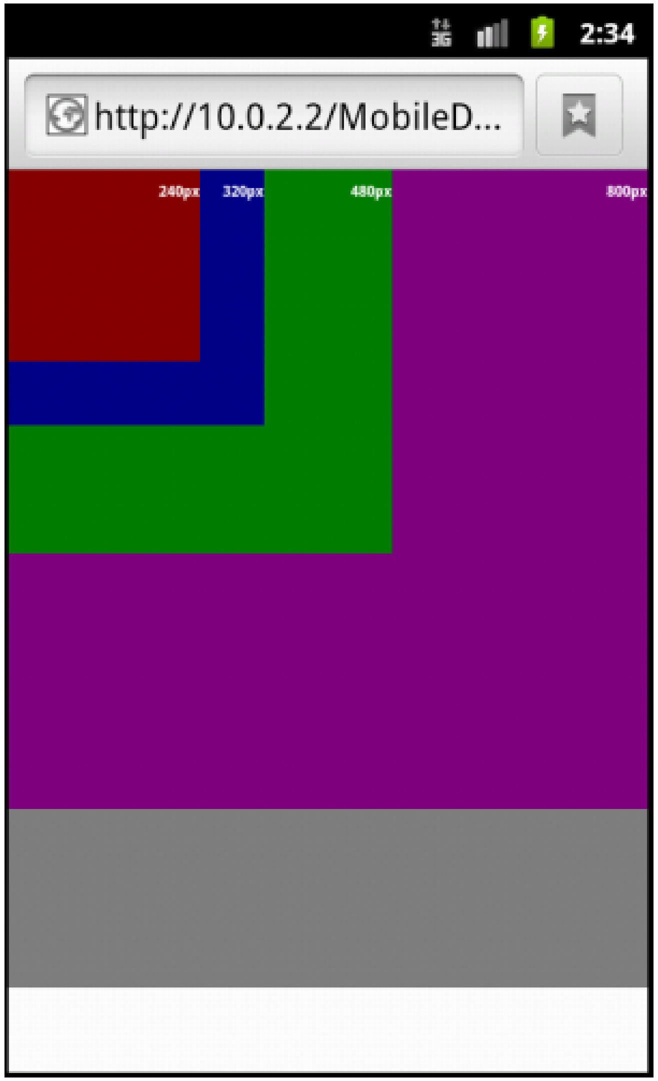
```
<body>
<div style="position:absolute;top:0px;left:0px;w
    <p style="text-align:right" class="auto-styl
</div>
<div style="position:absolute;top:0px;left:0px;w
    <p style="text-align:right" class="auto-styl
</div>
<div style="position:absolute;top:0px;left:0px;w
    <p style="text-align:right" class="auto-styl
</div>
<div style="position:absolute;top:0px;left:0px;w
    <p style="text-align:right" class="auto-styl
</div>
<div style="position:absolute;top:0px;left:0px;w
    <p style="text-align:right" class="auto-styl
</div>
```


Web - C:₩inetpub₩wwwroot₩MobileDevExample₩04_adapting₩viewport_sizetest.html - Aptana Studio 3
File Edit Navigate Search Project Run Commands Window Help
viewport_sizetest.html
Preview: viewport_sizetest.html
file:///C:/inetpub/wwwroot/MobileDevExample/04_adapting/viewport_sizetest.html
240px
320px
480px
800px
1024px

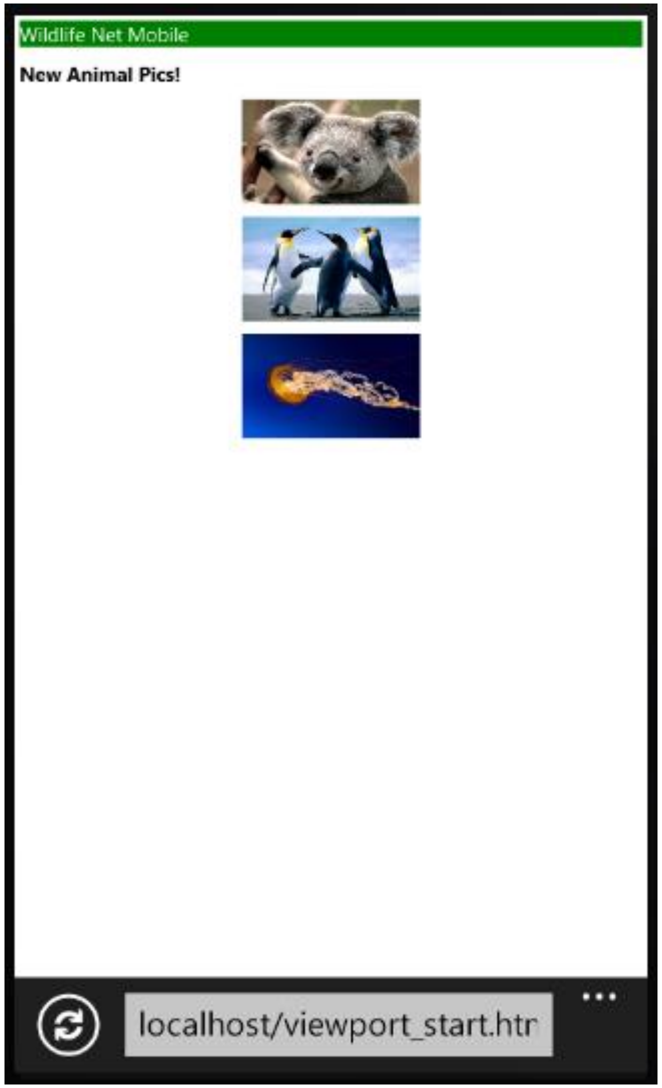
Display result: divs with varying widths: 240, 320, 480, 800, 1024 pixels

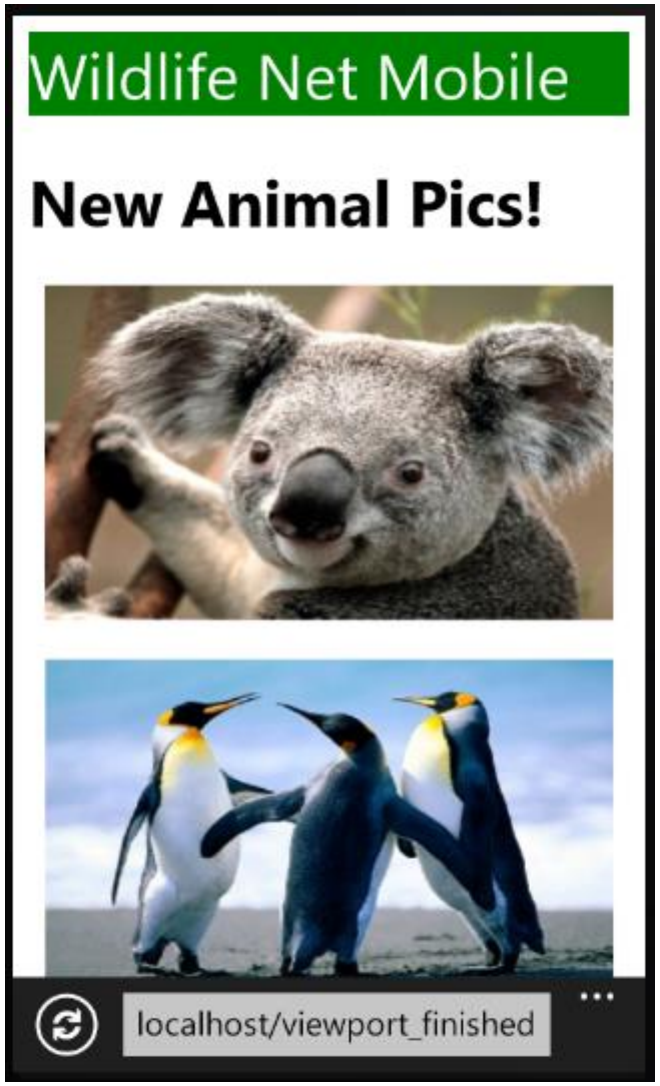
## Windows Phone vs. Android emulator

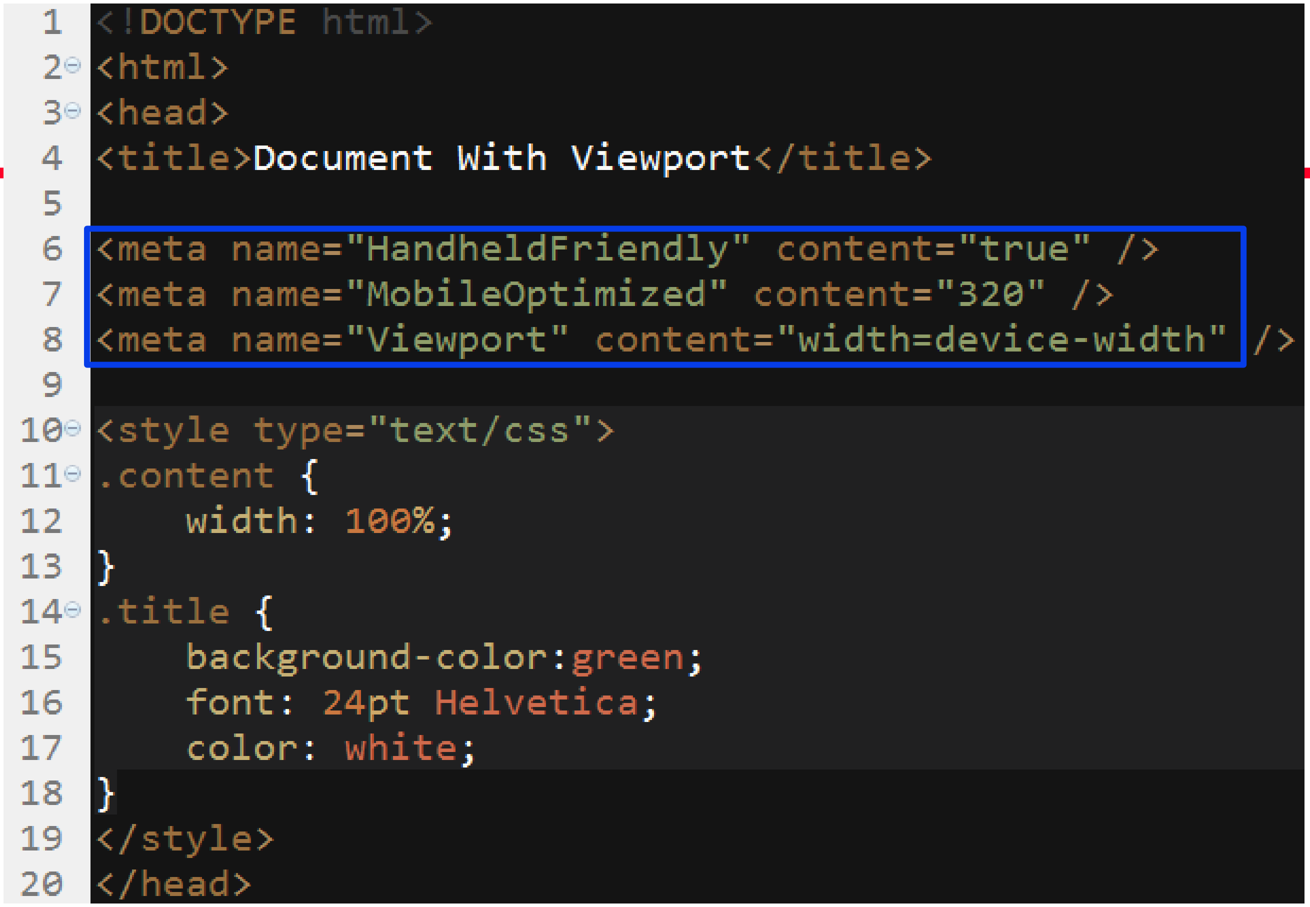




In no viewport setting


Wildlife Net Mobile
New Animal Pics!
localhost/viewport_start.htr


Wildlife Net Mobile
New Animal Pics!
localhost/viewport_finished



```
<!DOCTYPE html>
<html>
<head>
<title>Document With Viewport</title>

<meta name="HandheldFriendly" content="true" />
<meta name="MobileOptimized" content="320" />
<meta name="Viewport" content="width=device-width" />

<style type="text/css">
.content {
    width: 100%;
}
.title {
    background-color:green;
    font: 24pt Helvetica;
    color: white;
}
</style>
</head>
```

Modern browsers are smart enough to choose whichever tag is the most relevant one, the most recently invented, and ignore the ones they don't know about

## Contents

1. Understanding Content Adaptation Approaches
2. Controlling Viewport Layout
3. Using CSS Media Queries
4. Designing Forms

## Using CSS media queries

- Provide a way of selectively applying style sheets based upon certain characteristics of the media where the content will be displayed
  - Window size, screen size, orientation(portrait or landscape), color depth, resolution, etc.
- **W3C Recommendation 19 June 2012**
  - http://www.w3.org/TR/css3-mediaqueries/

## Commonly used CSS media query features

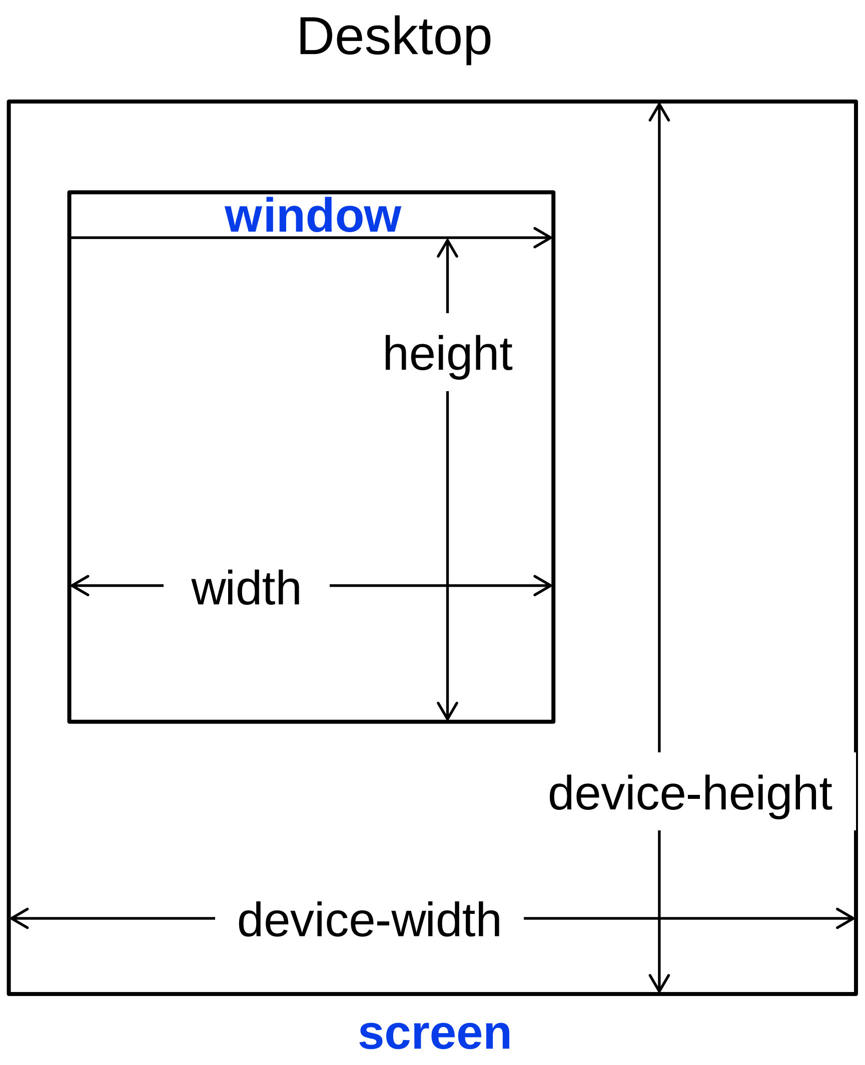
| Media Query Feature | Description |
|---|---|
| width | Indicates the width of an output surface such as a window (px, cm, etc.) |
| height | Indicates the height of an output surface |
| device-width | Indicates a device that has exactly the given width, in given units |
| device-height | Indicates a device that has exactly the given height, in given units |
| orientation | Used with “portrait” or “landscape” for devices that can do both. |

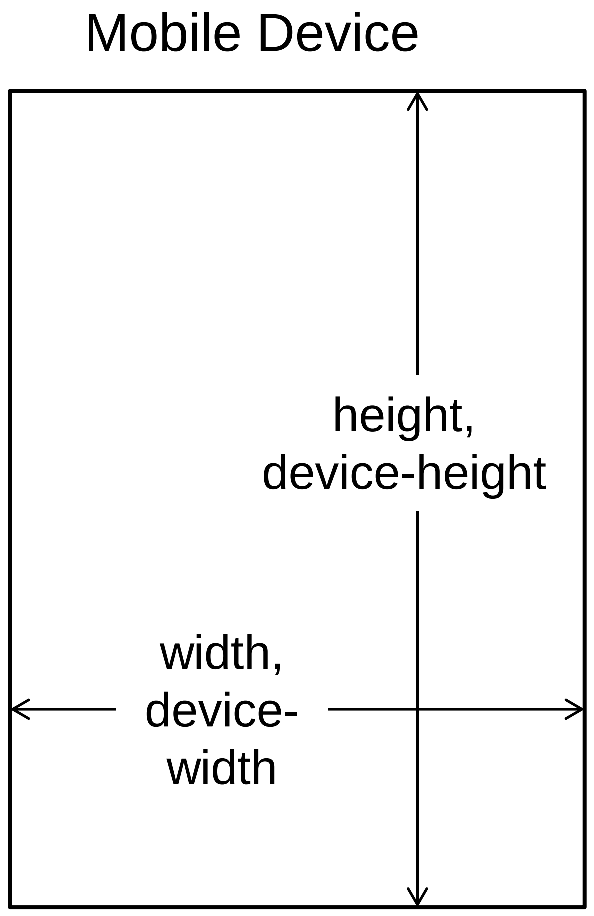
- For features that use units, you can specify any css-compliant unit, such as px, cm, in, em, etc.
- Many CSS-MQ features accept a “min” or “max” prefix
  - For example, you can use “min-width” or “min-device-width”
  - Works for all listed here except for “orientation”

| Media Query Feature | Description |
|---|---|
| aspect-ratio | Detects a display surface with a specific aspect ratio. typically used on TVs (4/3 for old TVs, 16/9 for HD TVs, etc) |
| device-aspect-ratio | Detects a device that has a specific aspect ratio |
| color | Detects a device with the given number of bits per color (1, 4, 8, etc) |
| color-index | Detects a device that uses a color lookup table instead of bits in each pixel |
| monochrome | Detects the number of bits per pixel for a monochrome device |
| resolution | Detects a device with the given resolution (in dpi or dpcm) |
| scan | Detects a device that uses progressive scanning for display (like a TV) |
| grid | Detects a device whose output is a bitmap or grid (like a tty device) |

- The min and max prefixes can be used with all of these features except for the “scan” and “grid” features.

# Using CSS media queries: desktop vs. mobile





## Defining CSS media queries

- Can define a media query directly within the <link> tag

  <!-- the stylesheet on screens where the window is up to 800px wide -->

  <link rel=“stylesheet” media=“screen and (max-width:800px)”
  	href=“example.css” />

- Or, within a CSS style sheet using @media

  @media screen and (min-width:801px) {

  /* style definitions that apply to screens greater than 801px wide */

  }

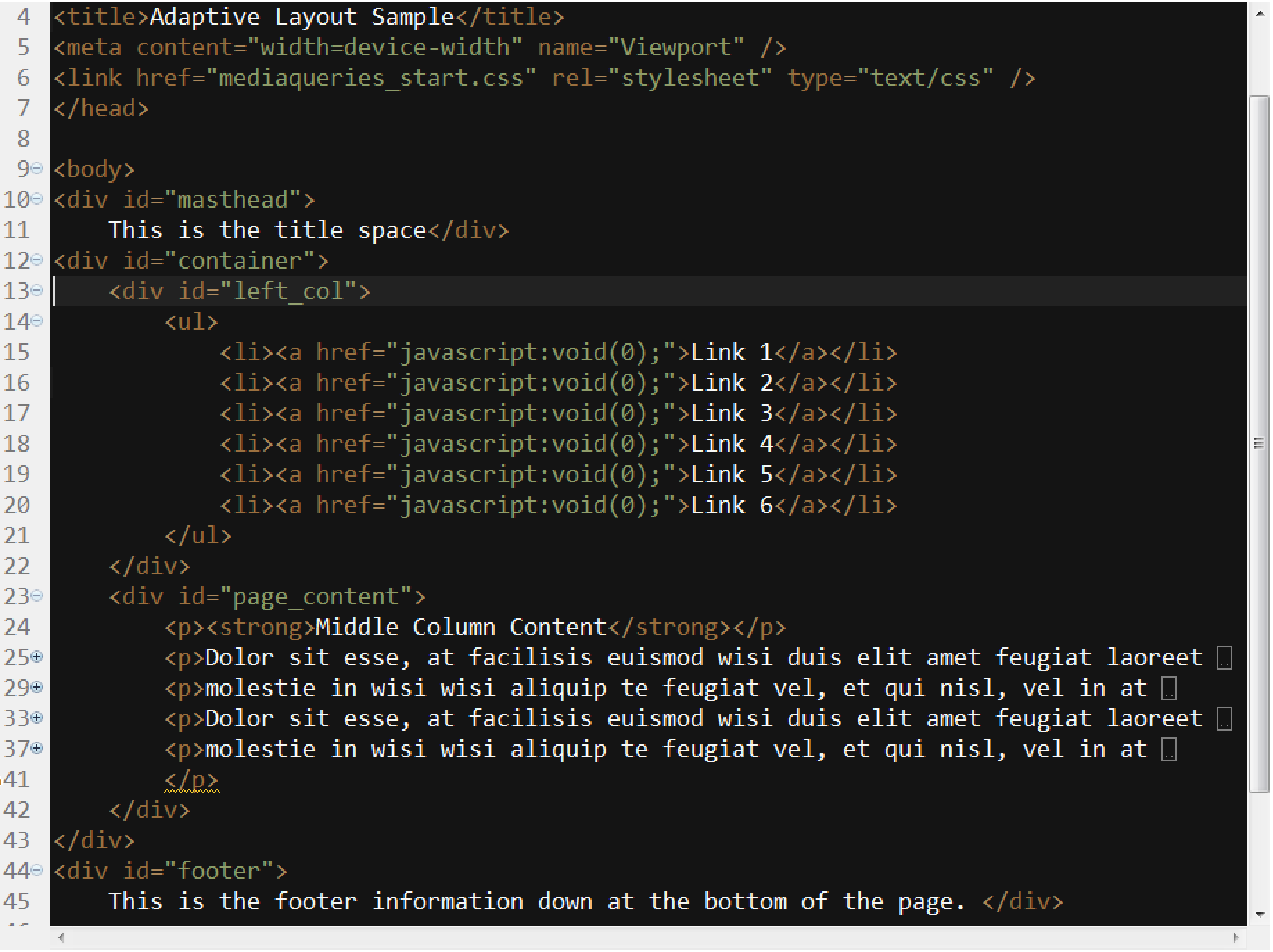
## Defining CSS media queries

- You can define multiple CSS Media Queries for different form factors

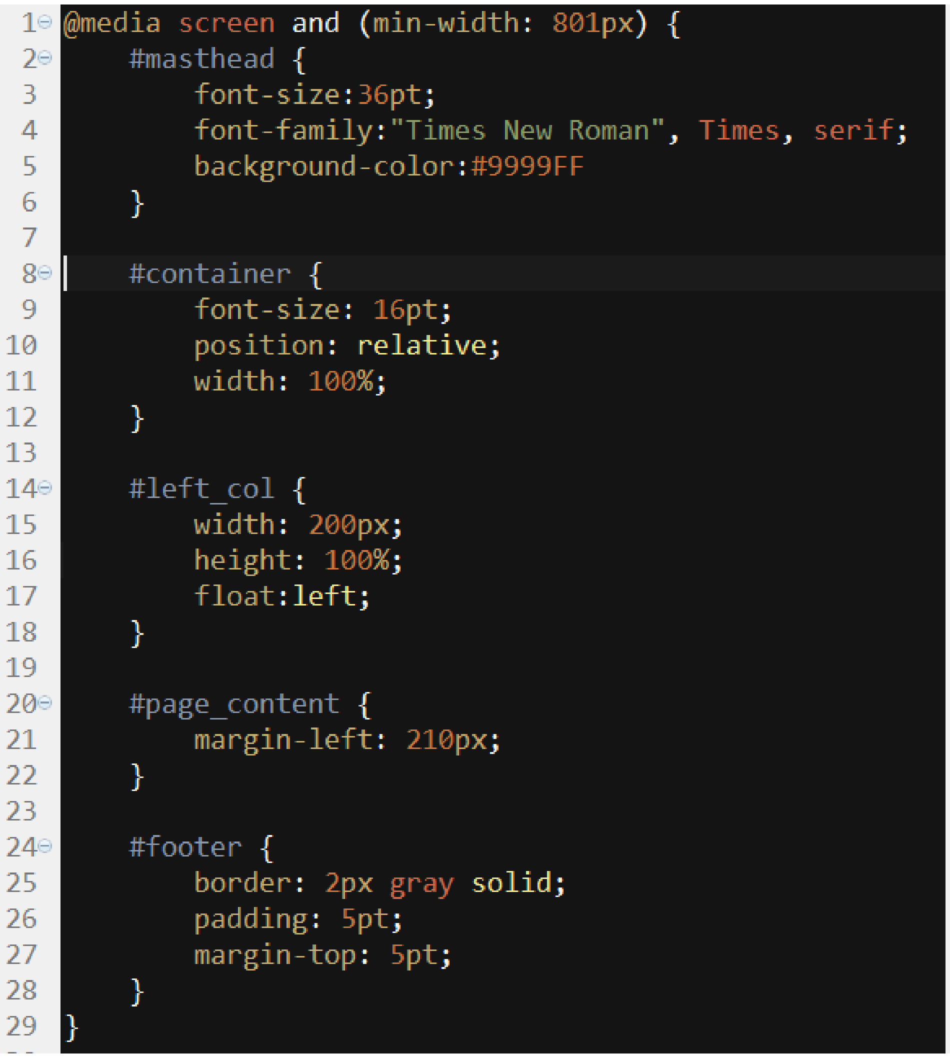
```
<!-- Styles to use for small screens, such as phones -->
<link rel="stylesheet" media="screen and (max-width:480px)" href="phone.css">

<!-- Styles to use for medium screens, such as tablets -->
<link rel="stylesheet" media="screen and (min-width:481px) and (max-
width:800px)" href="tablets.css">

<!--Styles to use for larger screens, such as laptops or desktops -->
<link rel="stylesheet" media="screen and (min-width:801px)" href="styles.css">
```

## Use width, or device-width?

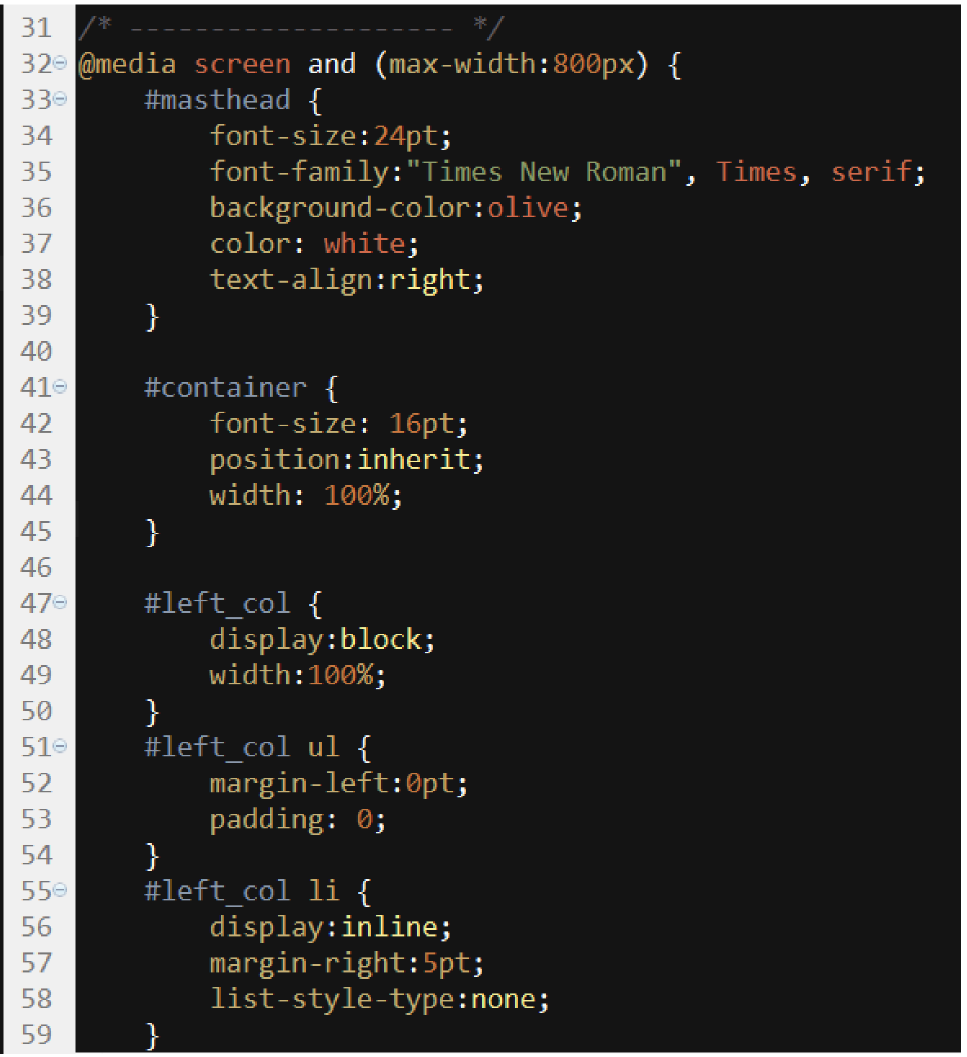
- The width and device-width properties mean slightly different things
  - The width property refers to the current display surface (window)
  - The device-width property refers to the screen display size
- General guidelines to follow:
  - If your style sheet will be used on both desktop and mobile devices, use the width property, since display surface (window) can change
  - If your style sheet will only be used on a mobile device, you can use the device-width property
- Usually, the width property is sufficient for most needs



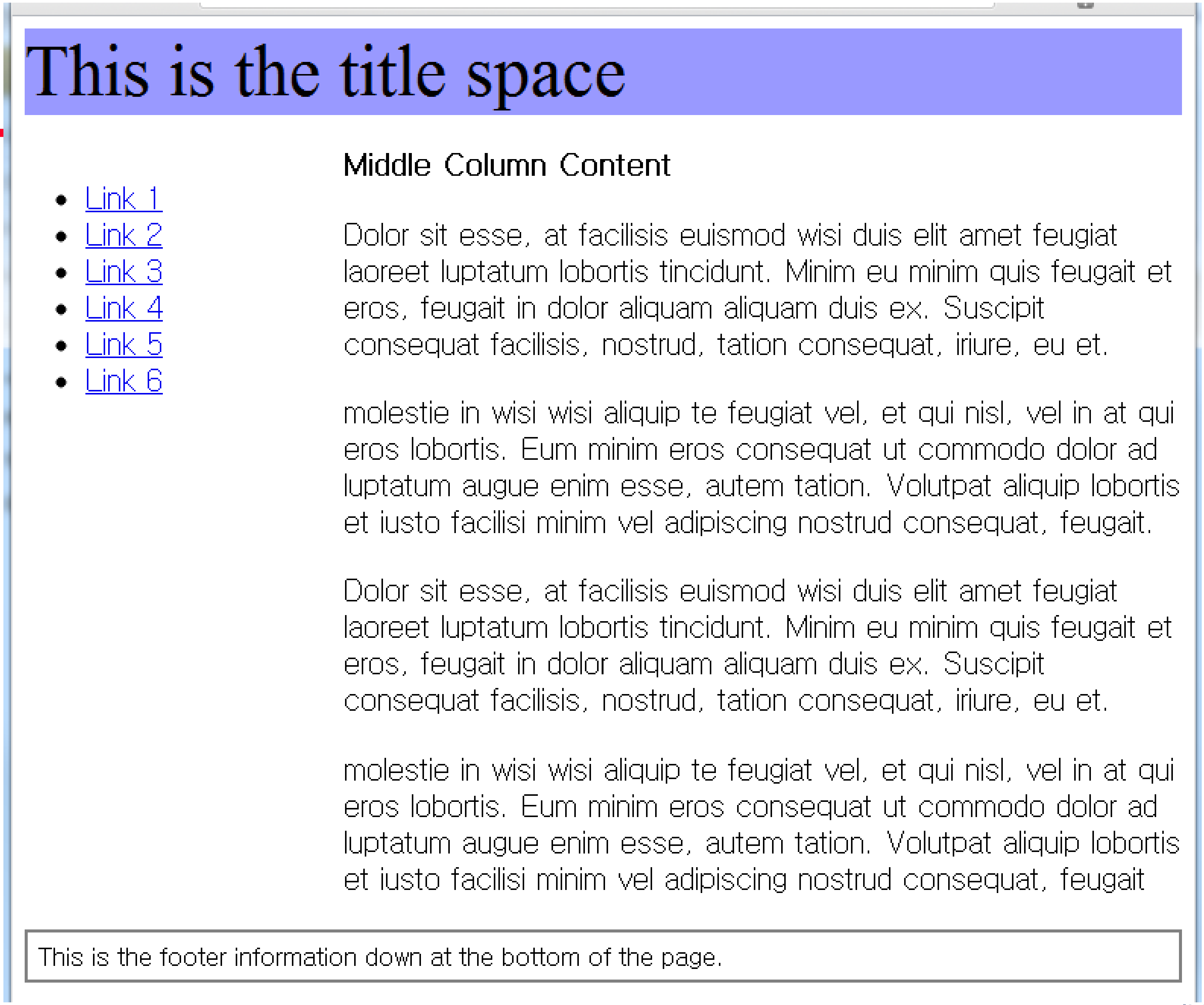
```
<title>Adaptive Layout Sample</title>
<meta content="width=device-width" name="Viewport" />
<link href="mediaqueries_start.css" rel="stylesheet" type="text/css" />
</head>

<body>
<div id="masthead">
    This is the title space</div>
<div id="container">
    <div id="left_col">
        <ul>
            <li><a href="javascript:void(0);">Link 1</a></li>
            <li><a href="javascript:void(0);">Link 2</a></li>
            <li><a href="javascript:void(0);">Link 3</a></li>
            <li><a href="javascript:void(0);">Link 4</a></li>
            <li><a href="javascript:void(0);">Link 5</a></li>
            <li><a href="javascript:void(0);">Link 6</a></li>
        </ul>
    </div>
    <div id="page_content">
        <p><strong>Middle Column Content</strong></p>
        <p>Dolor sit esse, at facilisis euismod wisi duis elit amet feugiat laoreet 
        <p>molestie in wisi wisi aliquip te feugiat vel, et qui nisl, vel in at 
        <p>Dolor sit esse, at facilisis euismod wisi duis elit amet feugiat laoreet 
        <p>molestie in wisi wisi aliquip te feugiat vel, et qui nisl, vel in at 
        </p>
    </div>
</div>
<div id="footer">
    This is the footer information down at the bottom of the page. </div>
```



```
@media screen and (min-width: 801px) {
    #masthead {
        font-size:36pt;
        font-family:"Times New Roman", Times, serif;
        background-color:#9999FF
    }

    #container {
        font-size: 16pt;
        position: relative;
        width: 100%;
    }

    #left_col {
        width: 200px;
        height: 100%;
        float:left;
    }

    #page_content {
        margin-left: 210px;
    }

    #footer {
        border: 2px gray solid;
        padding: 5pt;
        margin-top: 5pt;
    }
}
```



```
/* -------------------- */
@media screen and (max-width:800px) {
    #masthead {
        font-size:24pt;
        font-family:"Times New Roman", Times, serif;
        background-color:olive;
        color: white;
        text-align:right;
    }

    #container {
        font-size: 16pt;
        position:inherit;
        width: 100%;
    }

    #left_col {
        display:block;
        width:100%;
    }
    #left_col ul {
        margin-left:0pt;
        padding: 0;
    }
    #left_col li {
        display:inline;
        margin-right:5pt;
        list-style-type:none;
    }
```


This is the title space

- Link 1
- Link 2
- Link 3
- Link 4
- Link 5
- Link 6

Middle Column Content

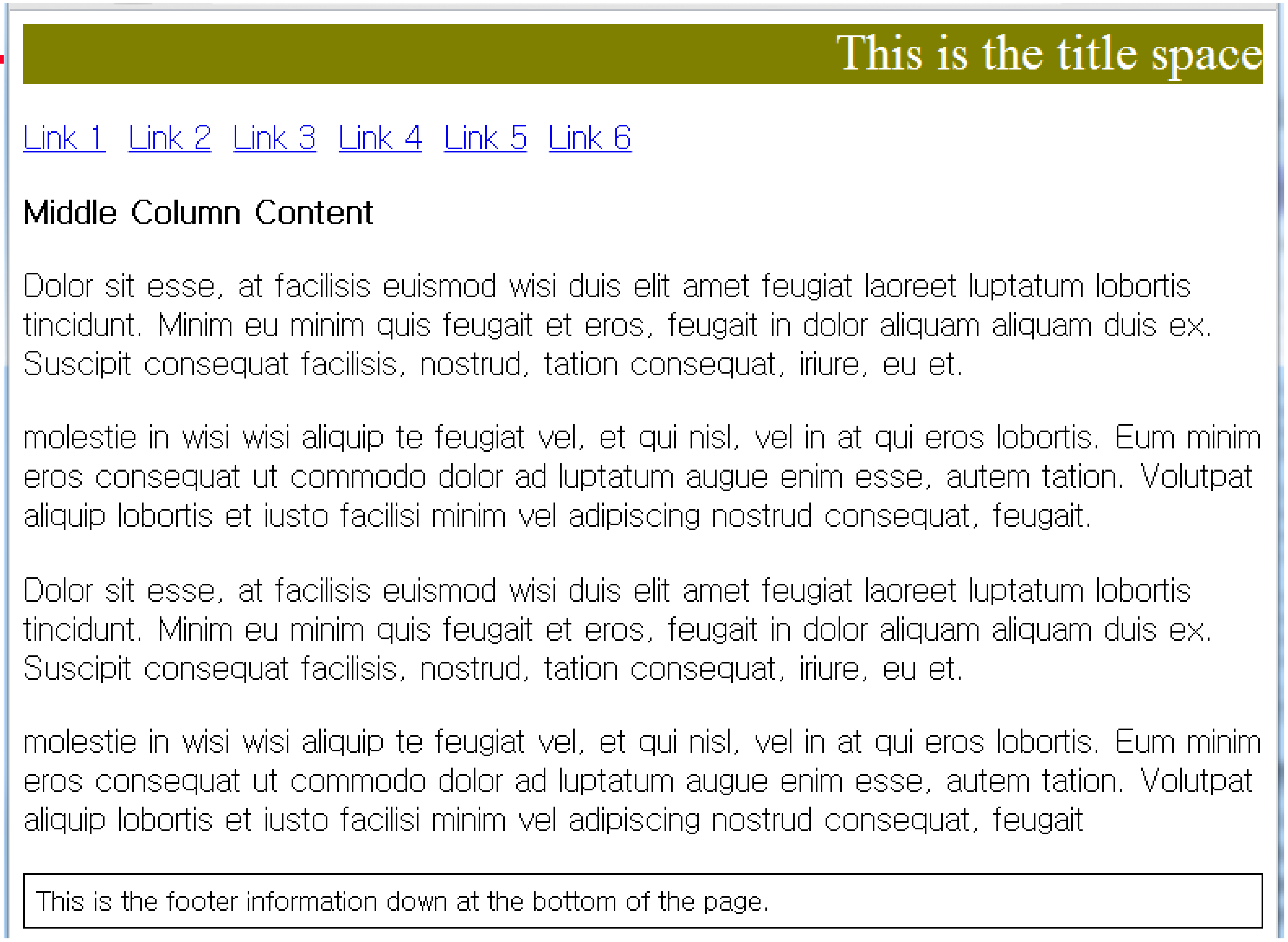
Dolor sit esse, at facilisis euismod wisi duis elit amet feugiat laoreet luptatum lobortis tincidunt. Minim eu minim quis feugait et eros, feugait in dolor aliquam aliquam duis ex. Suscipit consequat facilisis, nostrud, tation consequat, iriure, eu et.

molestie in wisi wisi aliquip te feugiat vel, et qui nisl, vel in at qui eros lobortis. Eum minim eros consequat ut commodo dolor ad luptatum augue enim esse, autem tation. Volutpat aliquip lobortis et iusto facilisi minim vel adipiscing nostrud consequat, feugait.

Dolor sit esse, at facilisis euismod wisi duis elit amet feugiat laoreet luptatum lobortis tincidunt. Minim eu minim quis feugait et eros, feugait in dolor aliquam aliquam duis ex. Suscipit consequat facilisis, nostrud, tation consequat, iriure, eu et.

molestie in wisi wisi aliquip te feugiat vel, et qui nisl, vel in at qui eros lobortis. Eum minim eros consequat ut commodo dolor ad luptatum augue enim esse, autem tation. Volutpat aliquip lobortis et iusto facilisi minim vel adipiscing nostrud consequat, feugait

This is the footer information down at the bottom of the page.



This is the title space

Link 1 Link 2 Link 3 Link 4 Link 5 Link 6

Middle Column Content

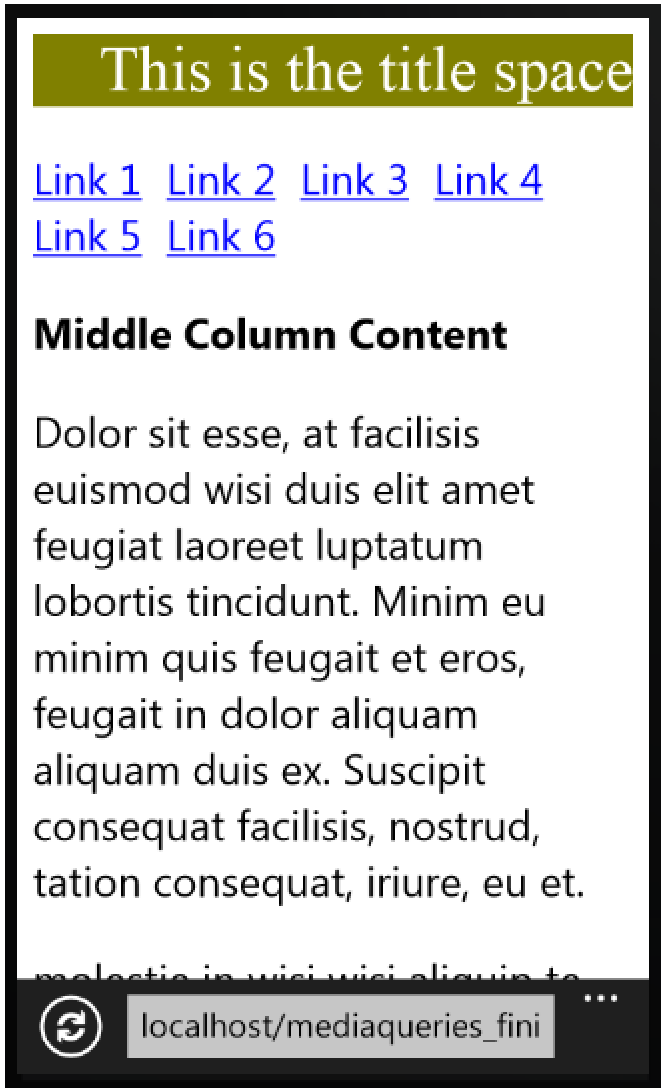
Dolor sit esse, at facilisis euismod wisi duis elit amet feugiat laoreet luptatum lobortis tincidunt. Minim eu minim quis feugait et eros, feugait in dolor aliquam aliquam duis ex. Suscipit consequat facilisis, nostrud, tation consequat, iriure, eu et.

molestie in wisi wisi aliquip te feugiat vel, et qui nisl, vel in at qui eros lobortis. Eum minim eros consequat ut commodo dolor ad luptatum augue enim esse, autem tation. Volutpat aliquip lobortis et iusto facilisi minim vel adipiscing nostrud consequat, feugait.

Dolor sit esse, at facilisis euismod wisi duis elit amet feugiat laoreet luptatum lobortis tincidunt. Minim eu minim quis feugait et eros, feugait in dolor aliquam aliquam duis ex. Suscipit consequat facilisis, nostrud, tation consequat, iriure, eu et.

molestie in wisi wisi aliquip te feugiat vel, et qui nisl, vel in at qui eros lobortis. Eum minim eros consequat ut commodo dolor ad luptatum augue enim esse, autem tation. Volutpat aliquip lobortis et iusto facilisi minim vel adipiscing nostrud consequat, feugait

This is the footer information down at the bottom of the page.



This is the title space

Link 1 Link 2 Link 3 Link 4
Link 5 Link 6

**Middle Column Content**

Dolor sit esse, at facilisis euismod wisi duis elit amet feugiat laoreet luptatum lobortis tincidunt. Minim eu minim quis feugait et eros, feugait in dolor aliquam aliquam duis ex. Suscipit consequat facilisis, nostrud, tation consequat, iriure, eu et.

localhost/mediaqueries_fini

## Contents

1. Understanding Content Adaptation Approaches
2. Controlling Viewport Layout
3. Using CSS Media Queries
4. Designing Forms

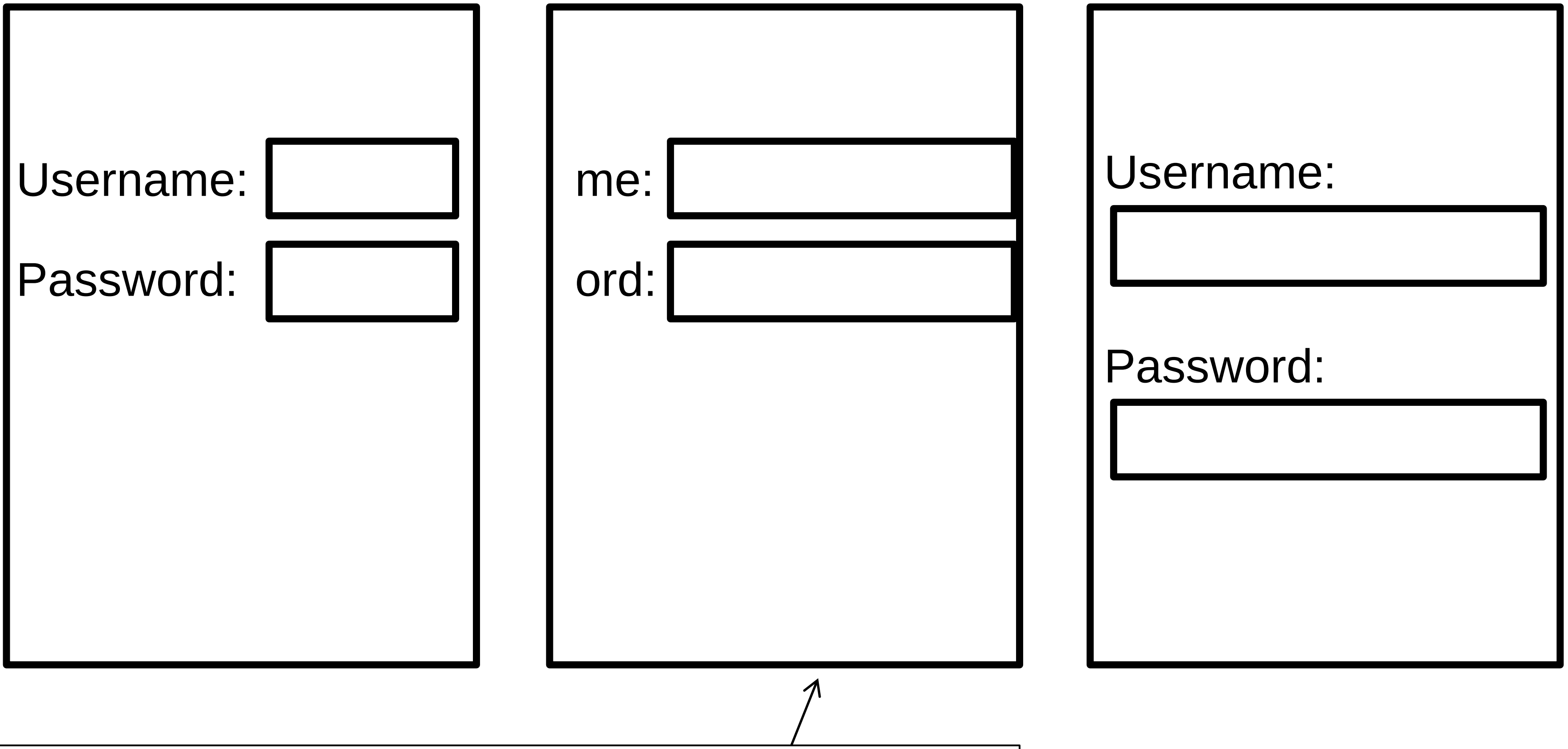
## Designing forms for mobile

- Forms on mobile devices are constrained by several factors
    - such as the width of the screen and the device’s input capabilities

## Designing forms for mobile

- Basic checklist for mobile Web forms:
  - Minimize the number of screens the user has to pass through
  - Use top-aligned form labels instead of left- (or right -) aligned fields
  - Give the user the option to display password characters
  - Use the new HTML5 Form input types

## Using top-aligned form field labels



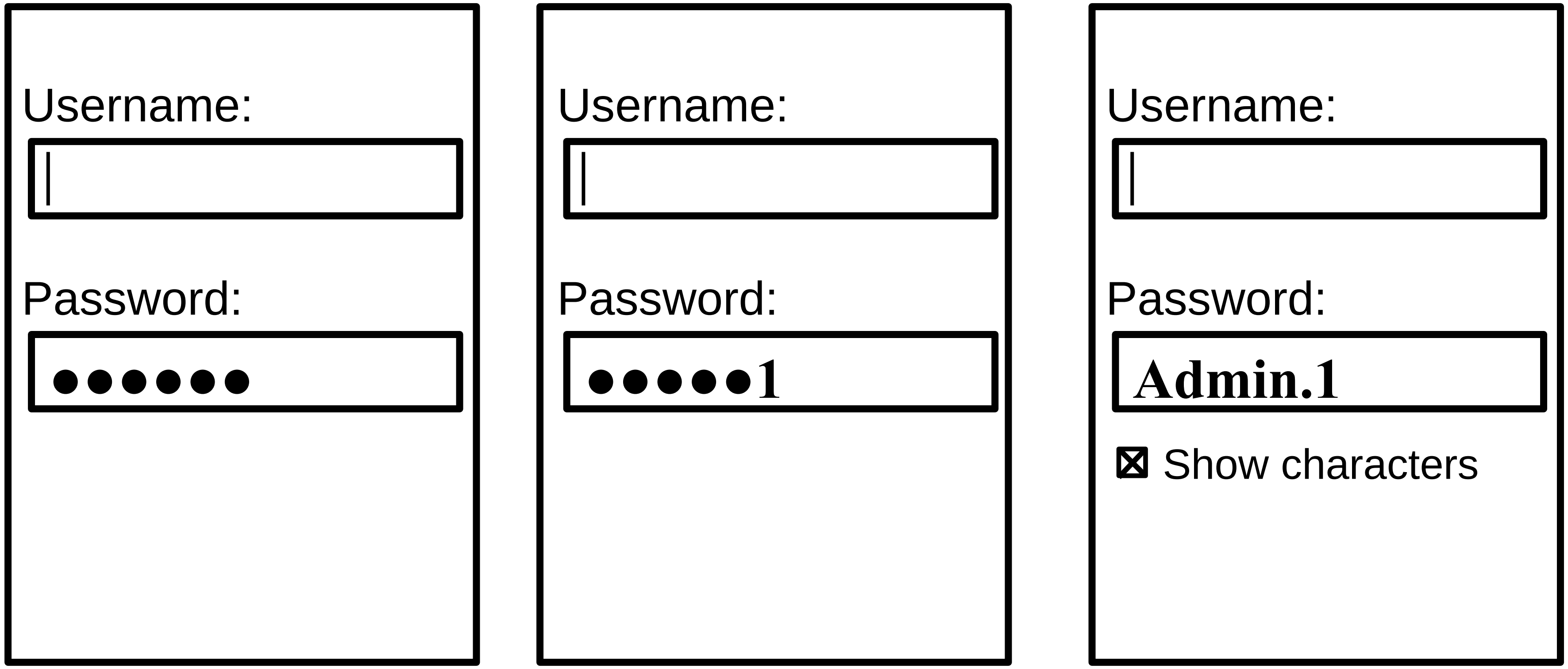
when the user puts the focus in the edit field, the user experience zooms in

## Using top-aligned form field labels: advantages

- Not need any fancy CSS or tables to lay out the form
- Labels stay visible when form is zoomed in
- Labels work well even when using left-to-right or right-to-left text
- Localization is easier : if field names grow when translated, they don't cause the form to re-flow (or at least flow vertically)
- Top-aligned labels lend themselves to vertical scrolling

## Handling password entry

- Give the user the option to show characters when entering passwords

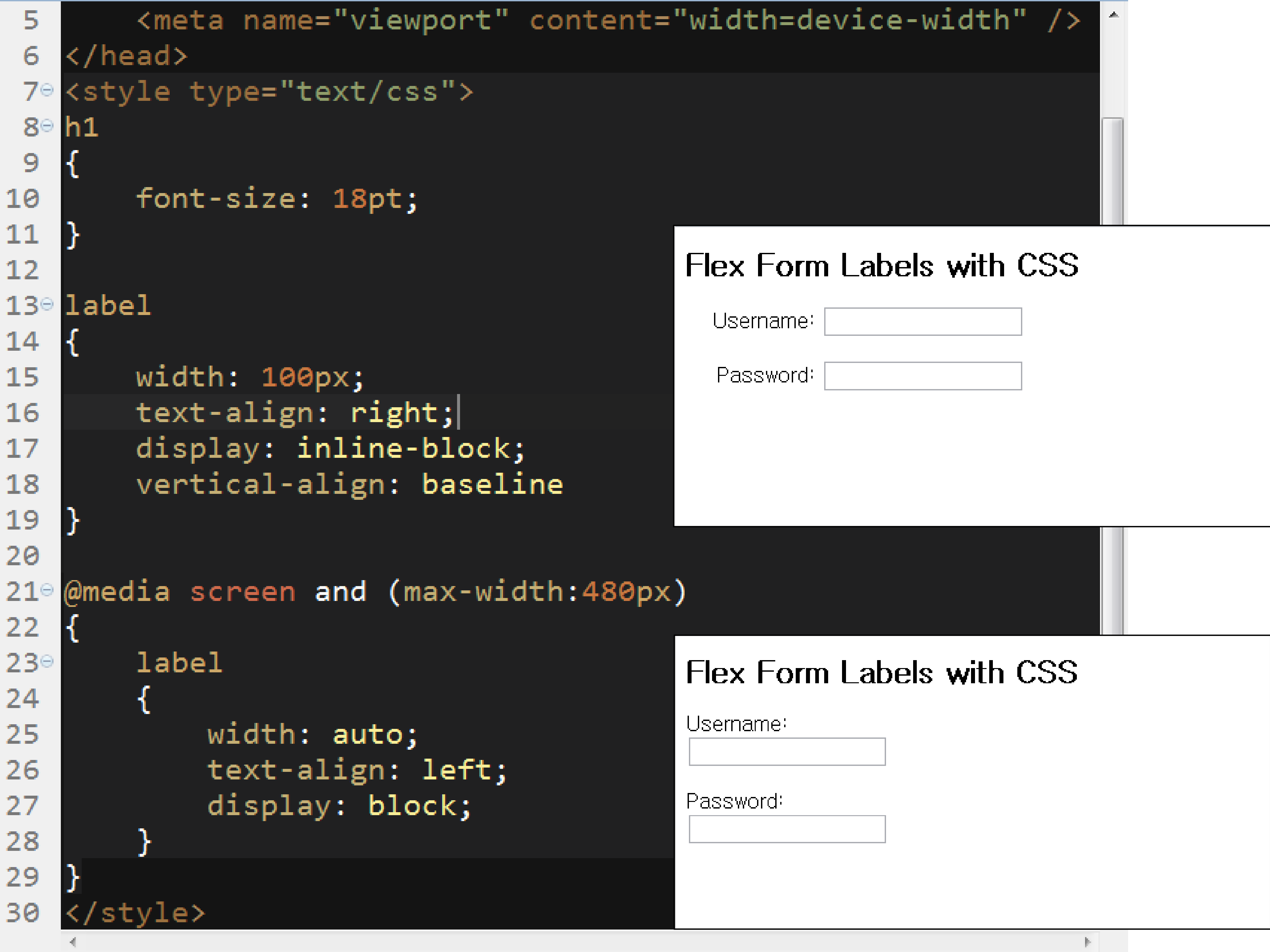


## Using the HTML5 form input types

- HTML5 defines more than a dozen new input types
  - You can use them right now. The new input types fall back to plain text fields in older browsers
- Two types of form enhancements:
  - Input types, which give a hint to the browser to display a more complex UI for certain kinds of fields
  - Input attributes, which provide enhanced interactivity or other functionality on the form field
    - Giving the form field the *autofocus*

## Using the HTML5 form input types

| Input Type | Description |
|---|---|
| email | Indicates a field that is an email address |
| url | Indicates a field that is a URL |
| number | Indicates a field that is intended for numerical input |
| range | Indicates a field that has a minimum and maximum range |
| color | Indicates a field that contains a color value |
| date | Indicates a field that contains a date or time |
| search | Indicates a field that is used to perform searching functions |



```
    <meta name="viewport" content="width=device-width" />
</head>
<style type="text/css">
h1
{
    font-size: 18pt;
}

label
{
    width: 100px;
    text-align: right;
    display: inline-block;
    vertical-align: baseline
}

@media screen and (max-width:480px)
{
    label
    {
        width: auto;
        text-align: left;
        display: block;
    }
}
</style>
```

Flex Form Labels with CSS

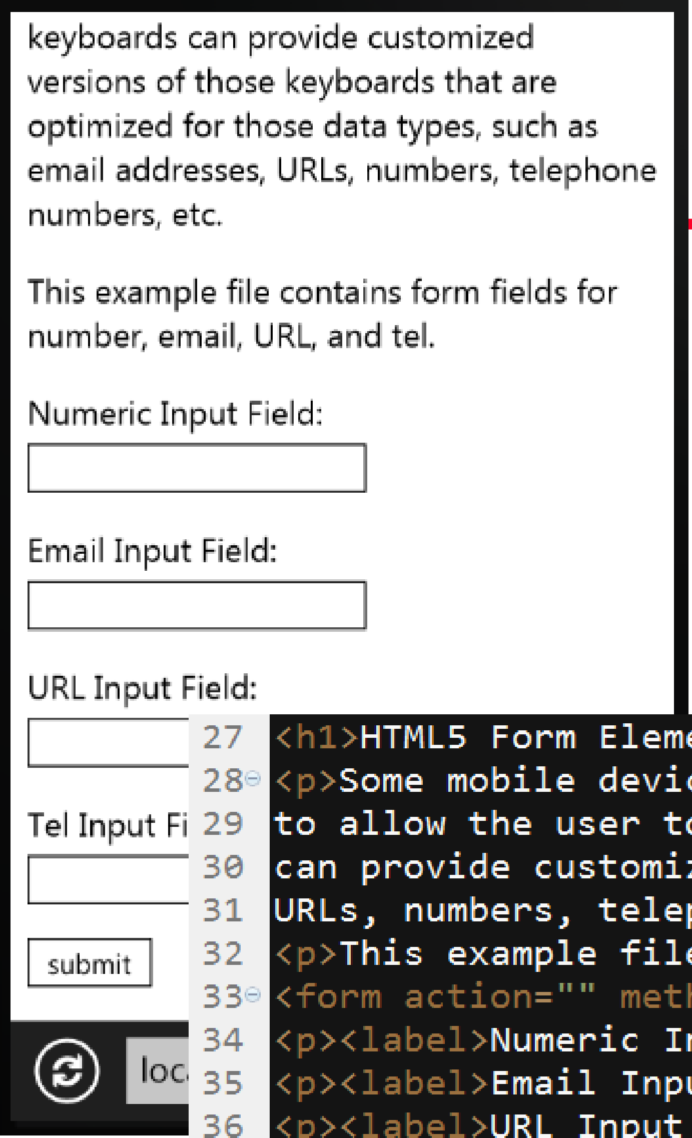
Username:

Password:

Flex Form Labels with CSS

Username:

Password:



keyboards can provide customized versions of those keyboards that are optimized for those data types, such as email addresses, URLs, numbers, telephone numbers, etc.

This example file contains form fields for number, email, URL, and tel.

Numeric Input Field:

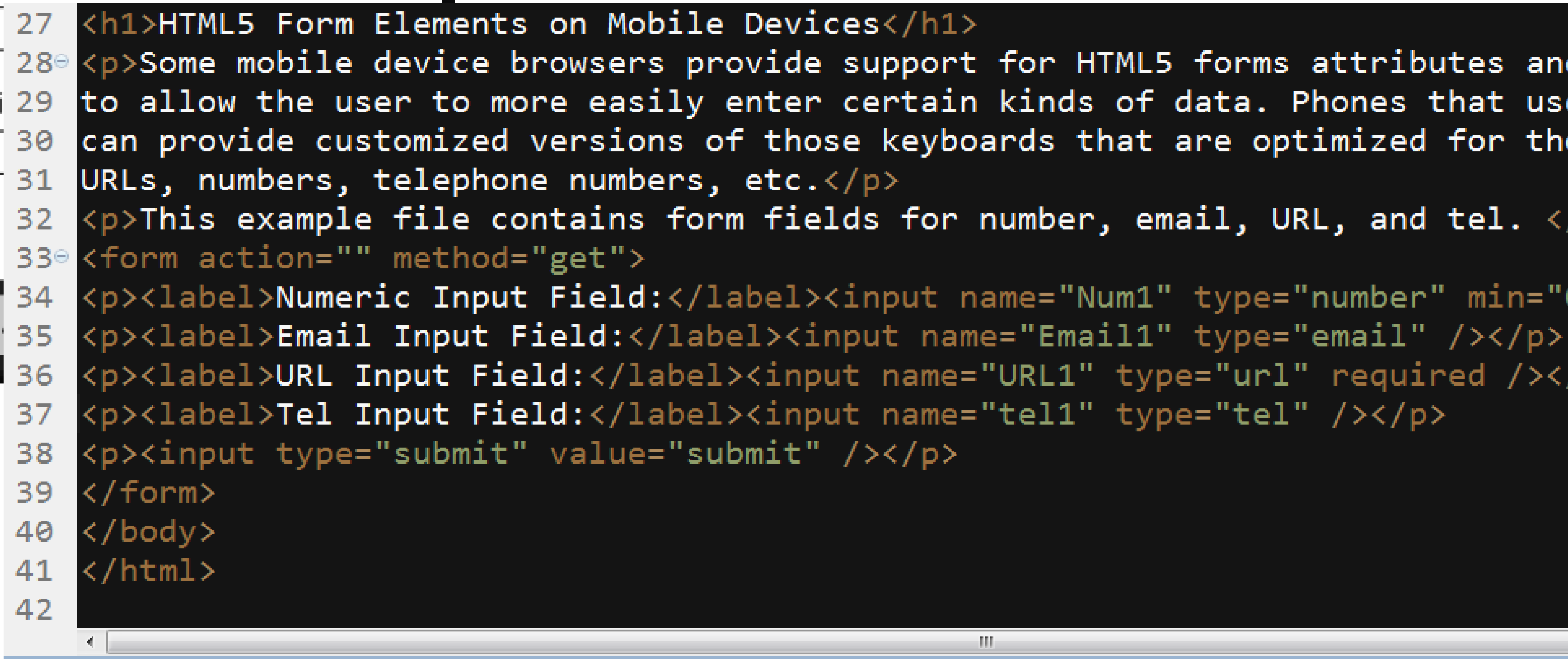
Email Input Field:

URL Input Field:

Tel Input Fi

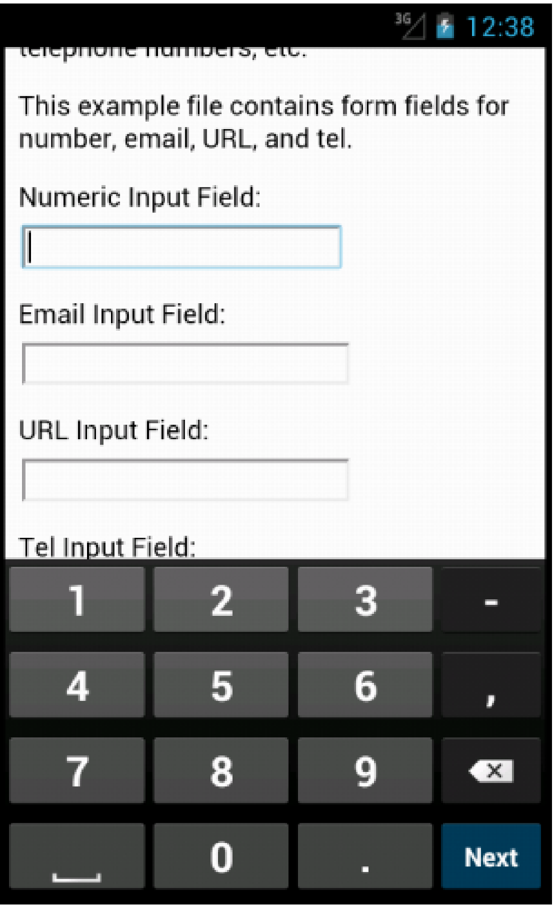
submit

loc



```
<h1>HTML5 Form Elements on Mobile Devices</h1>
<p>Some mobile device browsers provide support for HTML5 forms attributes an
to allow the user to more easily enter certain kinds of data. Phones that us
can provide customized versions of those keyboards that are optimized for th
URLs, numbers, telephone numbers, etc.</p>
<p>This example file contains form fields for number, email, URL, and tel. <
<form action="" method="get">
<p><label>Numeric Input Field:</label><input name="Num1" type="number" min="
<p><label>Email Input Field:</label><input name="Email1" type="email" /></p>
<p><label>URL Input Field:</label><input name="URL1" type="url" required /><
<p><label>Tel Input Field:</label><input name="tel1" type="tel" /></p>
<p><input type="submit" value="submit" /></p>
</form>
</body>
</html>

```

# keyboard that's optimized for input field


telephone numbers, etc.

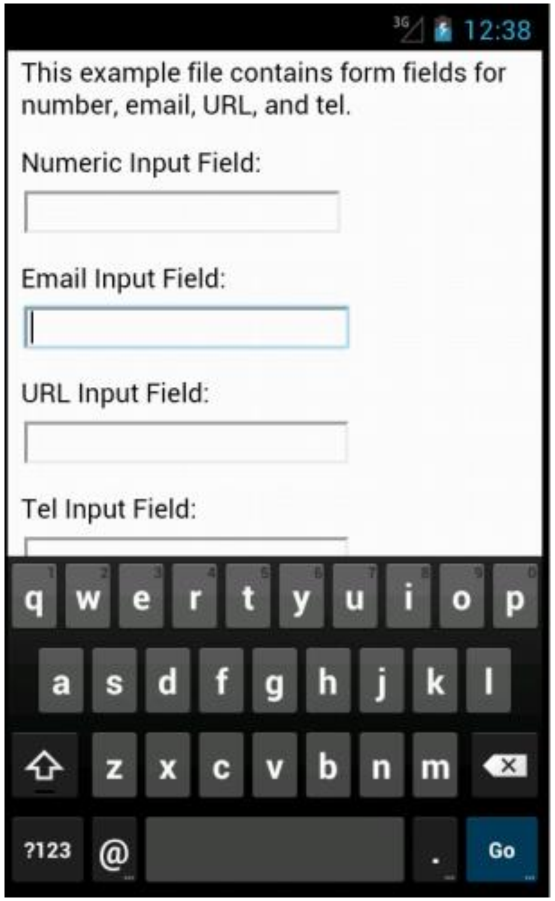
This example file contains form fields for number, email, URL, and tel.

Numeric Input Field:

Email Input Field:

URL Input Field:

Tel Input Field:

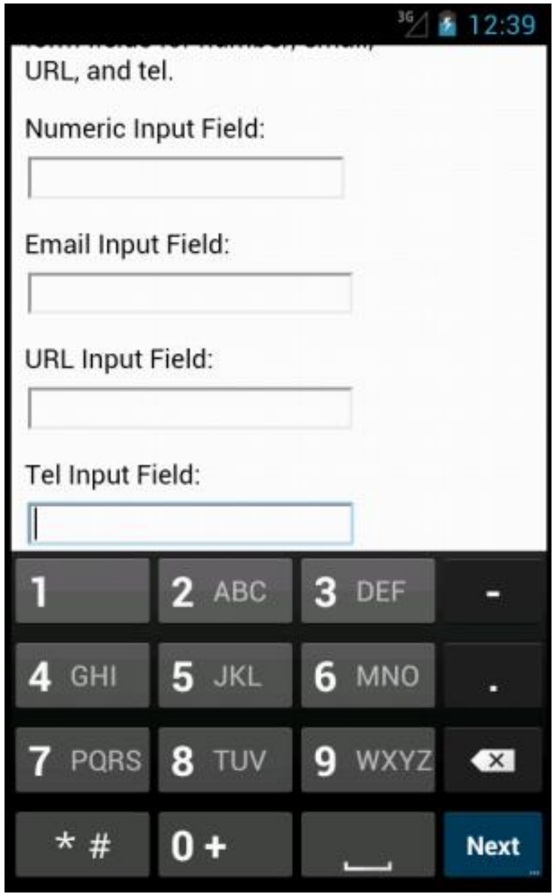

This example file contains form fields for number, email, URL, and tel.

Numeric Input Field:

Email Input Field:

URL Input Field:

Tel Input Field:


URL, and tel.

Numeric Input Field:

Email Input Field:

URL Input Field:

Tel Input Field: