UNIVERSITAS SCIENTIARUM SZEGEDIENSIS
UNIVERSITY OF SZEGED
Department of Software Engineering

Webfejlesztés Keretrendszerek - Bevezető

Dr. Bilicki Vilmos

Szoftverfejlesztés Tanszék

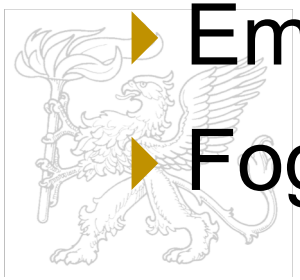
Összefoglaló

- Célok
- Követelmények
- Osztályzat
- Háttér
- Bevezető

Bemutatkozás

- Dr. Bilicki Vilmos
- Dugonics tér 150-es szoba
- Telefon: 6781-es mellék (mobil: +36203133523)
- Web: http://www.inf.u-szeged.hu/~bilickiv
- Email: bilickiv@inf.u-szeged.hu

- Fogadóóra: Hétfő 9-10



Követelmények, osztályzat

- Vizsga a vizsgaidőszakan a tételsorból két kérdésre kell írásban kifejtős választ adni.

Célok

- Stabil Angular alapok
- Minimális webes háttér ismeretek

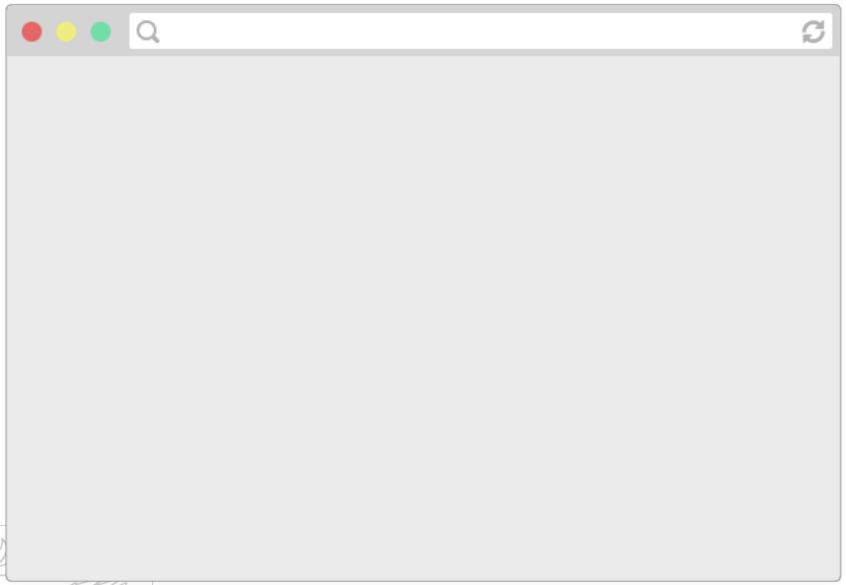
Nem cél

- HTML/CSS ismeretek
- Mély háttérrendszer ismeret
- JS/TS

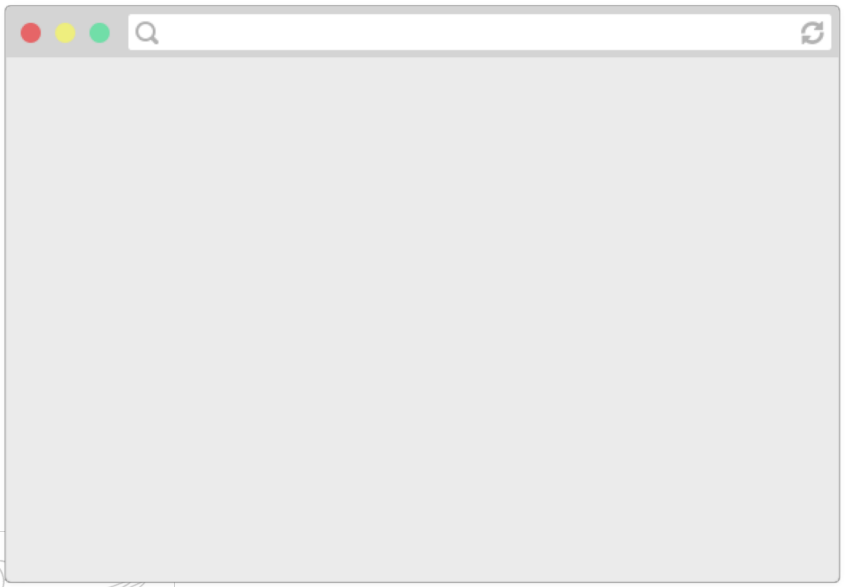
A félév áttekintése

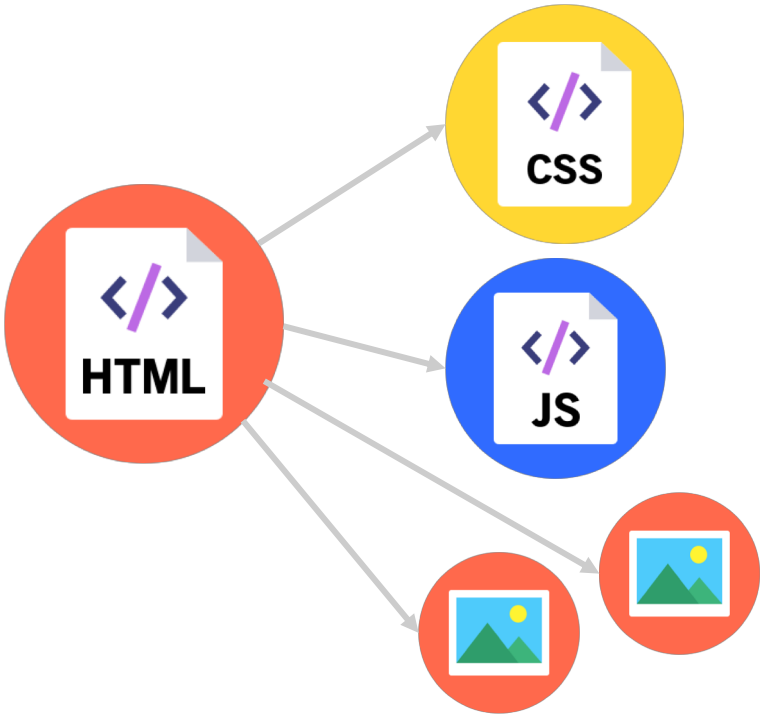
- Bevezető
- Angular Architektúra
- Angular Komponensek, sablonok
- Angular Űrlapok
- Angular NgModules
- Angular DI
- Angular HTTP
- Angular Routing,
- Navigation
- Angular Biztonság
- Angular Tesztelés
- Szerver oldali kereterendszerek: Node.JS
- Szerver oldali kereterendszerek: PHP

Hogyan működnek a weboldalak?

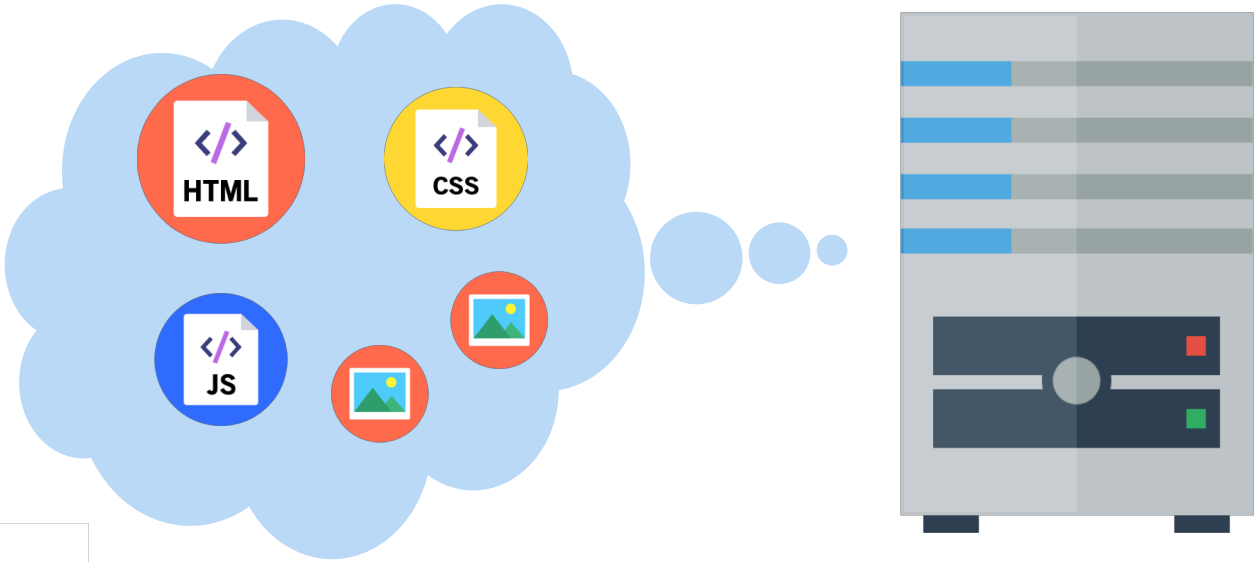


Hogyan működnek a weboldalak?

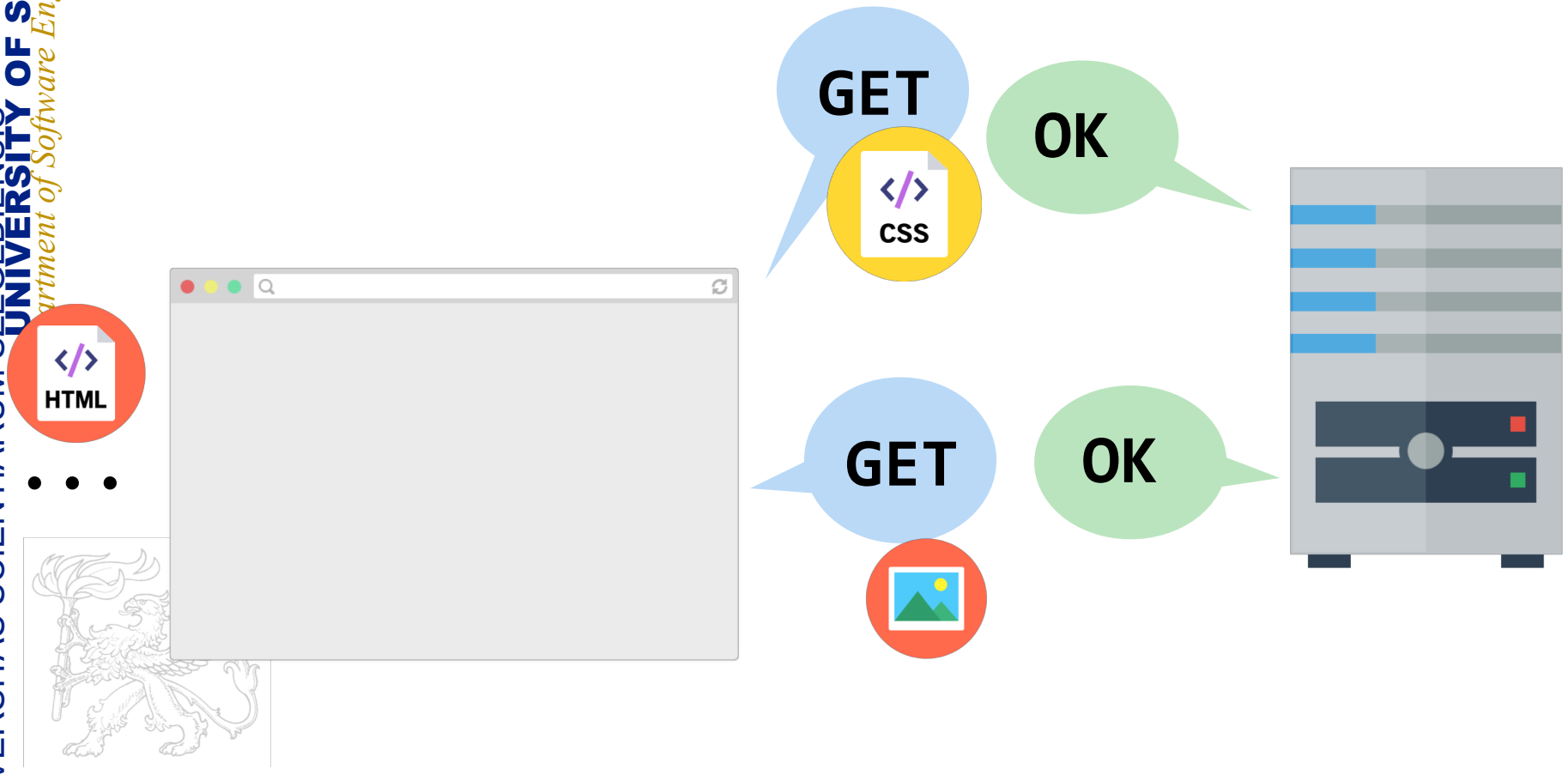




Hogyan működnek a weboldalak?



Hogyan működnek a weboldalak?



A HTML múltja, jelene

Tim Berners-Lee

- 1989: Létrehozták World Wide Web-et
  (WWW: a web odlalakat és a kiszolgáló protokollokat HTTP/HTTPS)
- 1994: Létrehozták a World Wide Web Consortium-ot
  - "W3C": Célja a web szabványok fejlesztése
  - Több nyelvet is kezel:
    - HTML, CSS, DOM, XML, etc
- 1997: Publikálták a "HTML4”-et
  - A HTML első stabil verziója

Kíméletes tönkremenetel

A W3C HTML szabvány több tervezési szempontot is felsorol, az egyik a kíméletes tönkermentel:

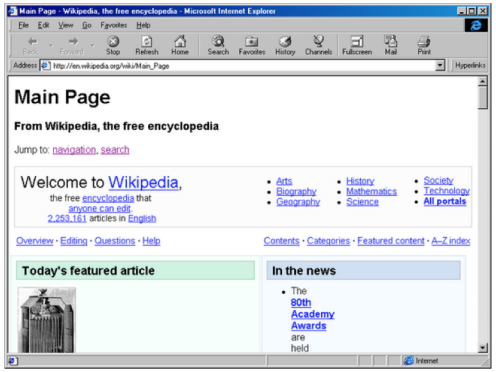
”Egy mozgólépcső sohasem mehet tönkre: csak lépcsővé válhat"

Ez az elv az ami miatt a web böngészők a legjobb szádnékkal próbálják értelmezni a nem érvényes HTML és CSS kódokat.

Legjobb szándék szerinti megjelenítés

A világa helyzete: 1997

A szabvány egy dolgot ír le



A böngésző egy másik dolgot csinál,

A fejlesztők furcsa nem szabványos dolgokat csinálnak

2000-es évek

Bocsi, a "Kíméletes tönkremenetel" hiba volt.

A böngészők ne jelentíség meg azokat az oldalakat amelyek nem szabványos HTML-ek

W3C

(Ez volt az XHTML 1.1 javaslat)

2000-es évek

W3C

Igen, természet esen

Mi?!?!

Nem tehetjük ezt!

Ez tönkreteszi az Internetet!

2004 megalakult a WHATWHG

Égessünk fel mindent és kezdjük újra XHTML 1.1-el

(64 million weboldal)

Dolgozzunk a HTML5-ön

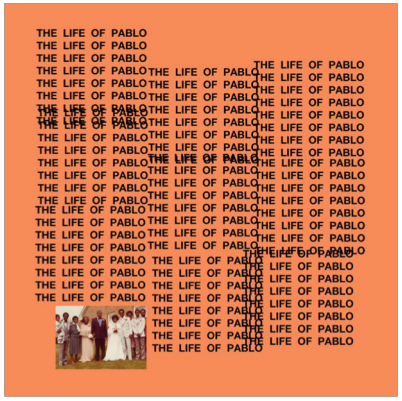
(egy nem tökéletes, de realisztikus szabványon)

Ugrás 2017-be?

- A W3C feladta az XHTML 1.1-et 2007-ben
- A W3C és a WHATWG csaknem barátok

HTML5 vs HTML

- W3C karbantartja a HTML5-öt:
  - Stabilizálja a WHATWG HTML-jét
  - Általában lemásolja a WHATWG amikor a por felszáll
- WHATWG karbantartja HTML-t: Élő Szabvány
  - Nincs sorszám, nincs verzió
  - Gyarkan firssül!
  - A böngészők a WHATWG-t valósítják meg



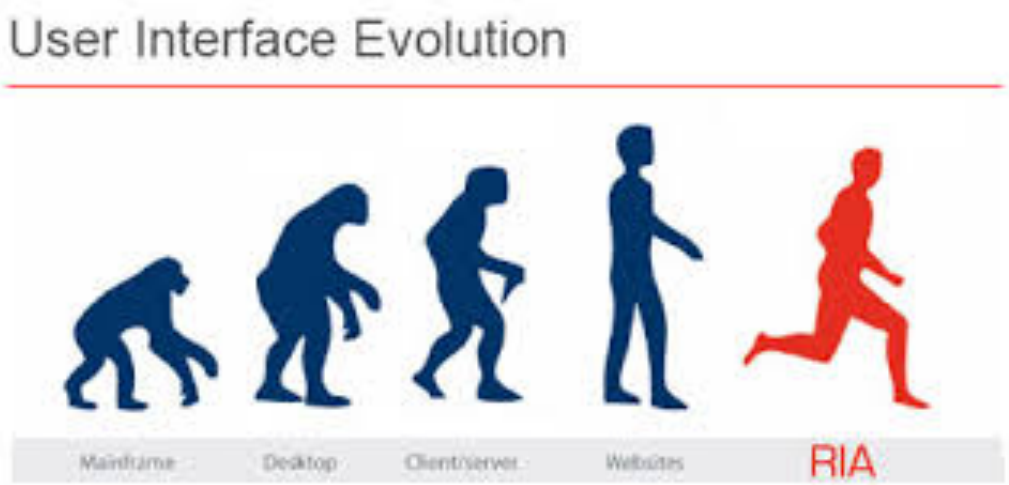
K: Milyen HTML elemeket használhatunk?

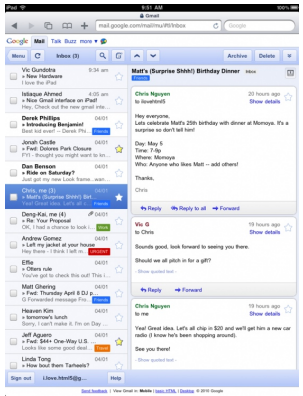
- Nézzük meg MDN HTML címke listáját

K: Honnan tudhatom, hogy melyik böngésző milyen (HTML/CSS/JS) elemet valósít meg?

- Nézzük meg a caniuse.com oldalt.

Gazdag Internet Alkalmazás

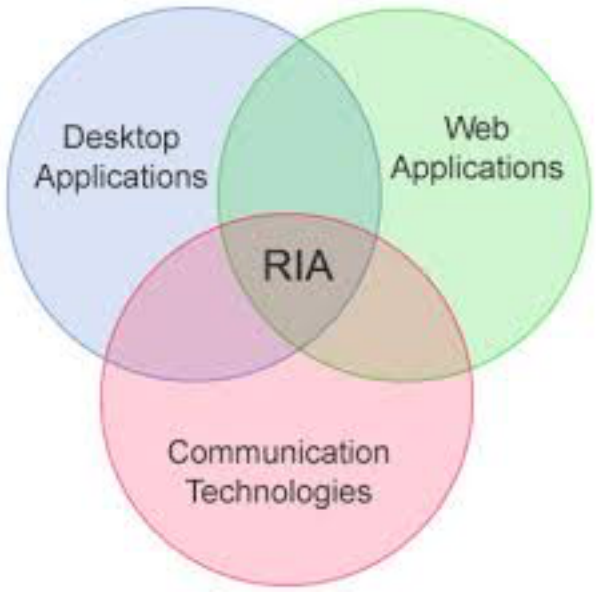


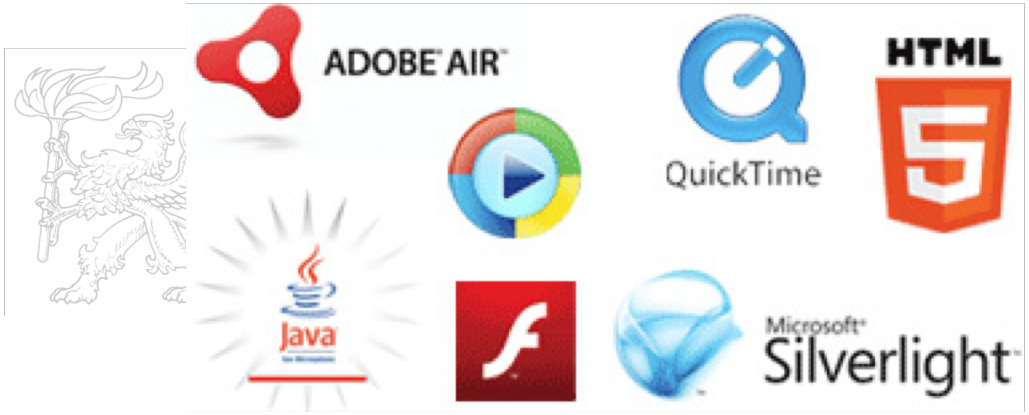


- Vékony kliens helyett vastag web kliens
- Web alkalmazás asztali alkalmzás érzéssel/képeséggel (TCO csökkentés)
  - Egyszerű olcsó telepítés karbantartás
  - Internet skála
- 2011-ig beépülő modulokkal

Gazdag Internet Alkalmazás

- Jellemzően böngésző beépülő modul alapú megoldás
- Asztali alkalmazás mint cél





Reszponzív Web Alkalmazás

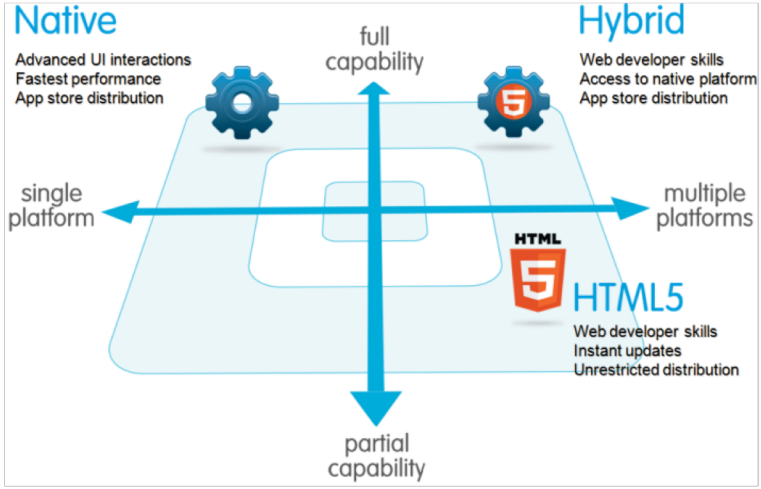
- Web alapú de mobilon futó alkalmazás mint cél
- A cél: mobil böngészőben is használható alkalmazás
- Böngésző alapú megoldás (JavaScript+HTML+CSS)
- 2001: audi.com (dinamikusan mértezhető weboldal)
- 2010-ben jelent meg mint jelentős igény

Reszponzív Web Alkalmazás

- A weboldal folyékony rács segítségével alkalmazkodik a felülethez (fluid grid)
  - Relativ pozícionálás
  - Lépték osztályok (sx, md, lg, ….)
- Flexibilis képek (mérthez alkalmazkodnak)
- Média lekérdezések
  - Más stílust használ más-más környezetben
- Technológia:
  - HTML5
  - AJAX
  - CSS (SASS, …)

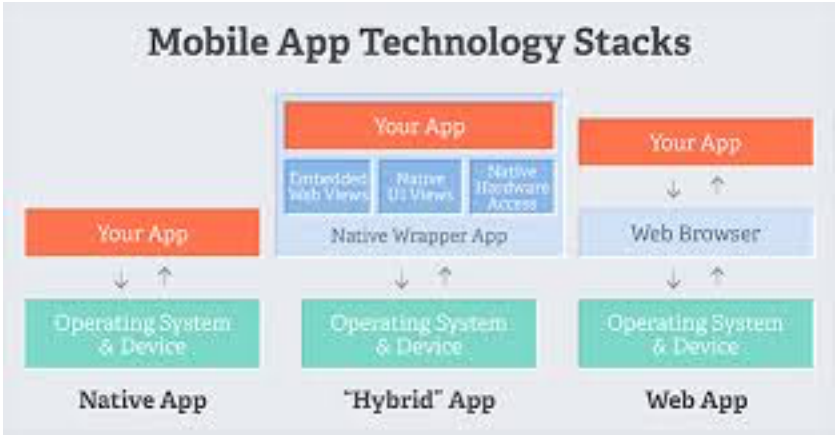
Hibrid mobil alkalmazások

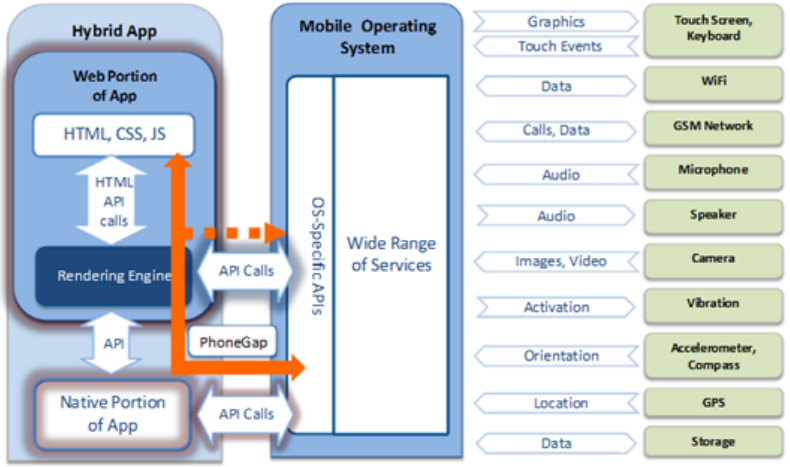
- Mobil alkalmazás képességek elérése mint cél
- Platform független
- Web szoftver verem
  - HTML5
  - AJAX
  - CSS
- Futtató környezet
  - Cordova



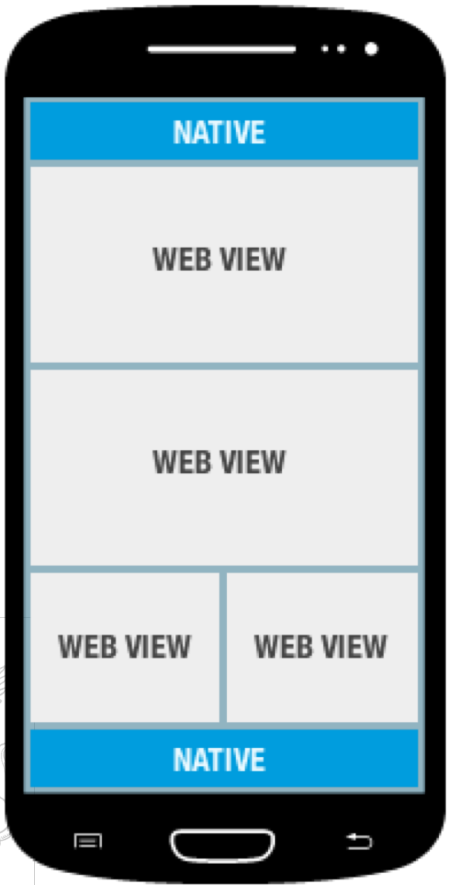
Hibrid mobil alkalmazások

- Egyedi elemek
  - GUI
  - Natív mobil
- Teljes fordító lánc (build chain)
- WebView mint futtató környezet
- Finom átmenet a mobil és a natív alkalmazás között
- Periféria támogatás
  - Okos óra, …





Hibrid mobil alkalmazások



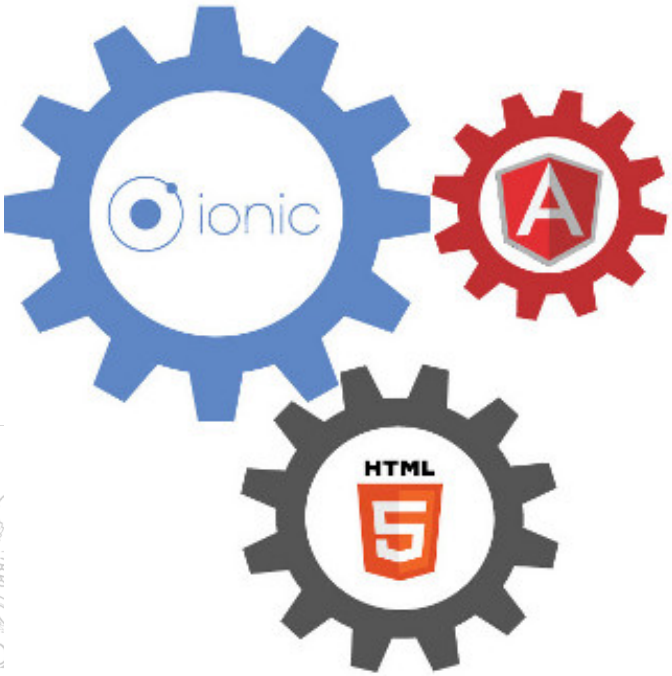
NATIVE CONTAINERS

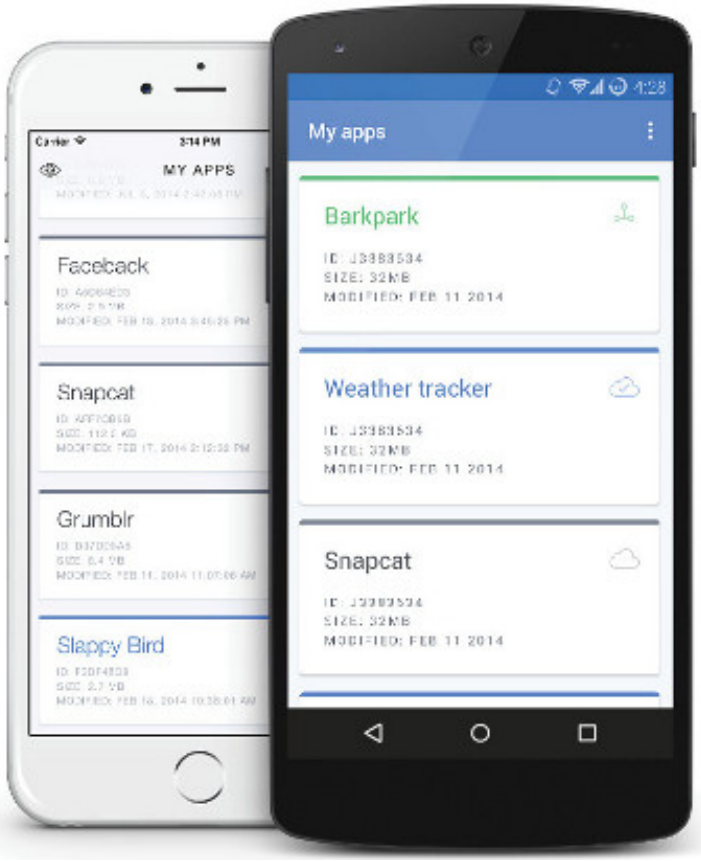
iOS -xCode/Objective-C
Android - Eclipse/Java
Windows - Visual Studio/C#

MOBILE WEB

HTML
CSS
JavaScript

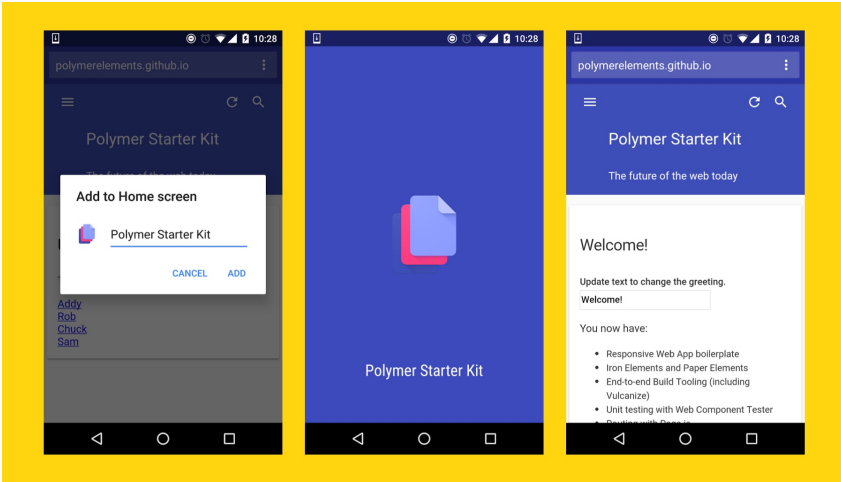
Hibrid mobil alkalmazások





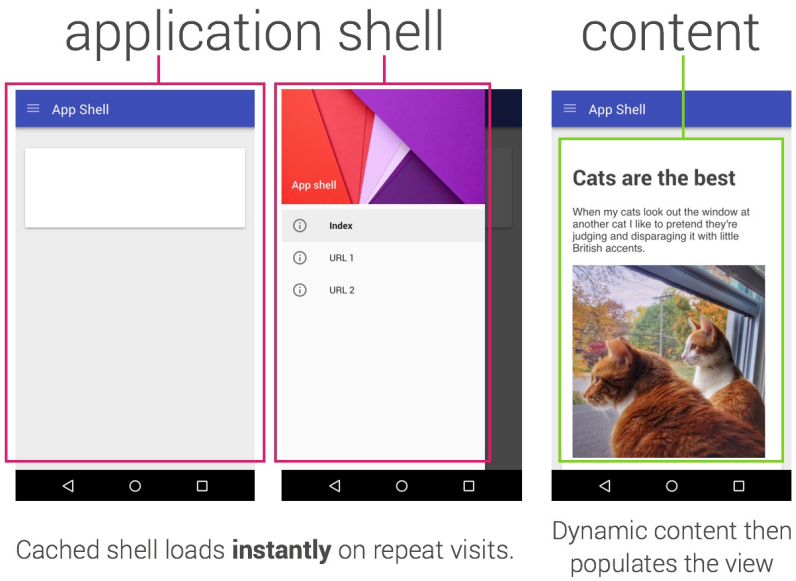
Progresszív Web Alkalmazás

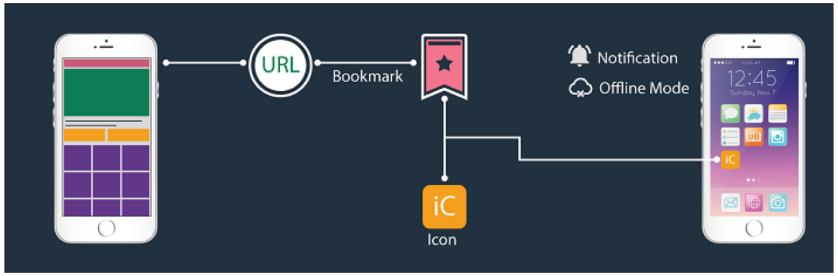
- Web képességekkel is bíró mobil alkalmazások
- Képességek:
  - **Progresszív** – Böngésző függetlenül működnek
  - **Reszponzív**- Felbontás függő működésmód
  - **Kapcsolat független** – A háttér folyamatok segítségével kezeli a lassú vagy adott időszakban hiányzó hálózati kapcsolatokat
  - **App szerű** – Ugyanúgy néz ki és működik mint a natív alkalmazás
  - **Friss** – Mindig naprakész (háttér frissítés szálak)
  - **Biztonságos** – HTTPS felett kommunikál
  - **Felderíthető** – W3C leírók segítségével a kereső motrok megismerhetik
  - **Motiváló** – Elérjük a felhazsnálót, pl.: push
  - **Telepíthető** – A felhazsnáló kihelyezheti a home képernyőre
  - **Hivatkozható** – URL segítségével megosztható, nem kell komplex telepítés



Progresszív Web Alkalmazás

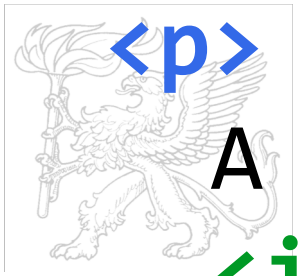
- Technológia:
  - HTML5
  - AJAX
  - CSS
  - Web leíró (W3C manifest)
  - Háttér folyamatok (worker)
  - App Shell
    - Cache API
    - Fetch API
    - IndexexDB API
    - Notification API
    - …





HTML blokk vs. inline

- HTML (Hypertext Markup Language)
  - A weboldal **tartalmát** és **struktúráját** írja le
  - Elemekből áll.
    - blokk: a tartalom nagyobb elemeivan szélességük és magasságuk
      <p>, <h1>, <blockquote>, <ol>, <ul>, <table>
    - inline: kis tartalom, nincs szélesség, magasság
      <a>, , ,
      - inline blokk: sorban tartalom amelynek van magassága, szélessége
        <img>
    - metaadat: az oldalról szóló információ, gyakran nem látható
      <title>, <meta>



```
<p>
  A HTML <em>nagyszerű!!!</em>
  <img src="puppy.png" />
</p>
```

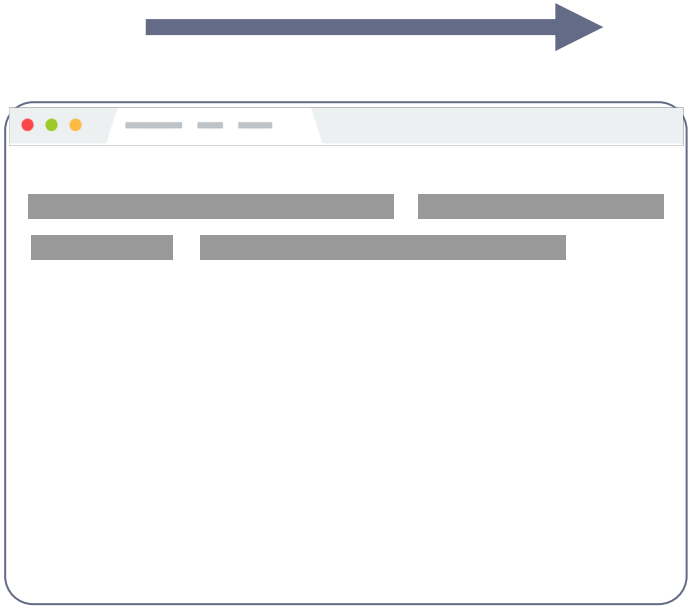
Blokk elemek

- Példák:
  <p>, <h1>, <blockquote>, <ol>, <ul>, <table>
- Az oldal tejles szélességét elfoglalják (fentről lefolyik)
- Van magasságuk és szelességük
- Lehetnek blokk és inline gyemrekeik

```
h1 {
  border: 5px solid red;
  width: 50%;
}
```

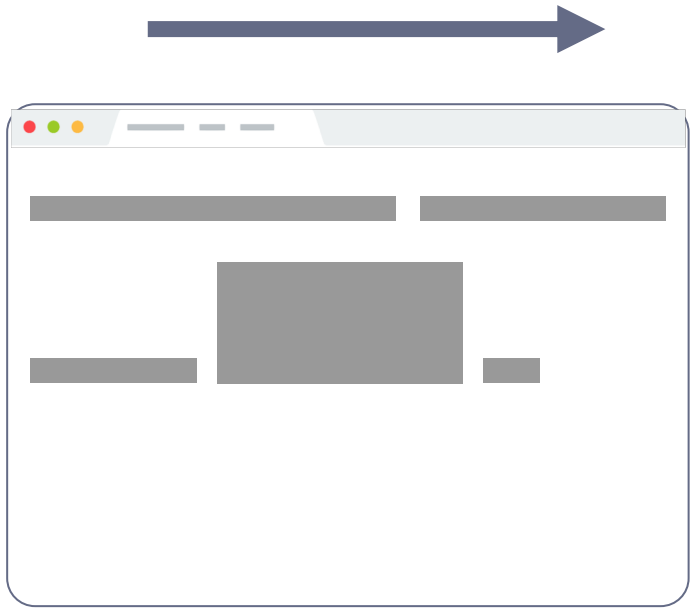
Inline elemek

- Példák:
  <a>, , ,
- Nincs magassága és szélessége
- Nem lehet blokk elem gyermeke
- Nem pozícionálható (pl.: CSS float és position)
- A tartalmazó blokk határozza meg a pozícióját



Inline blokk

- Példa: <img>, minden elem display: inline-block; tulajdonsággal
- A szélessége a tartalom szélessége (balról jobbra folyik)
- Lehet magassága és szélessége
- Lehet blokk elem gyermeke
- Lehet pozícionálni



CSS display tulajdonság

- Az adott elem megjelenítés típusát tudjuk vele megváltoztatni. Például:

```
p {
 display: inline;
}
```

```
a {
 display: block;
}
```

és

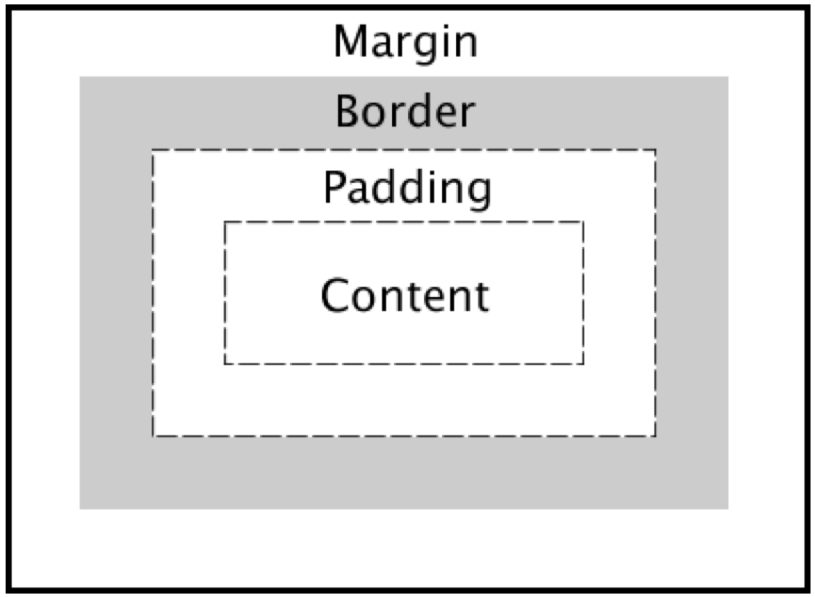
- Két általános elem, melynek nincs előre meghatározott célja, vagy stílusa:
  - : általános blokk elem
  - : általános inline elem

Helyettük

Név	Leírás
<p>	Paragrafus (mdn)
<h1>-<h6>	Szakasz fejrész (mdn)
<article>	Dokumentum, odal vagy site (mdn) Ez általában a body alatt van közvetlenül
<section>	A dokumentum általános szakasza (mdn)
<header>	Egy dokumentum bemutató szakasza(mdn)
<footer>	A dokumentum, vagy szakasz végén lévő lábléc(mdn)
<nav>	Navigációs szakasz (mdn)

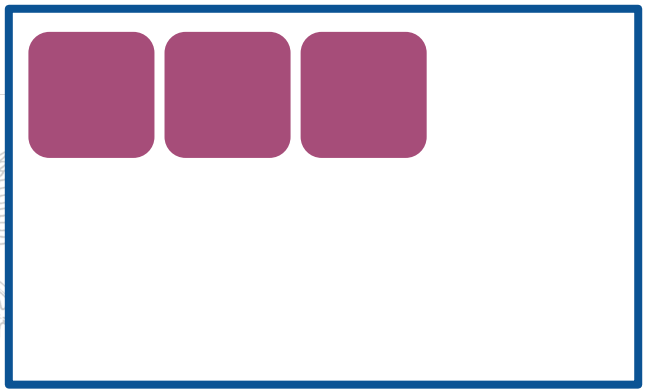
CSS doboz modell

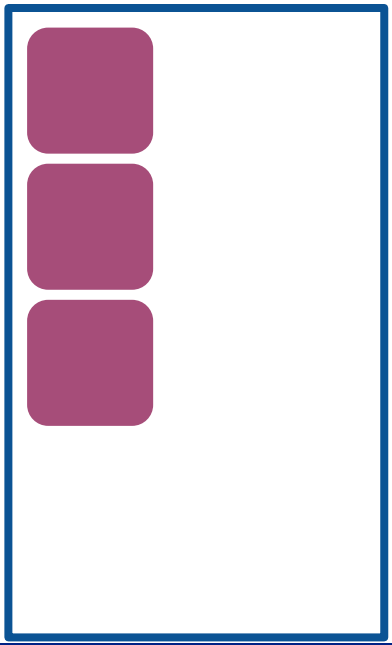
- Minden elem 4 rétegből áll
  - Az elem tartalmából
  - A tartalom körüli határvonalból
  - Az elemek közötti kitöltésből (belső)
  - A határvonal körüli helyből (külső)



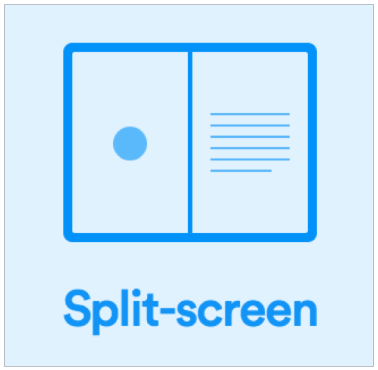
Flexbox modell

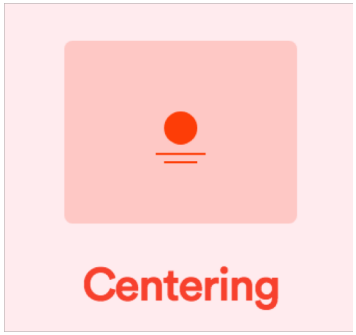
- Komplex képernyők létrehozására ahol nem elegendő a blokk és inline pozícionálás
- CSS megjelenítési mód: **Flex layout**
- Oszlopkba és sorokba rendezi az elmeket

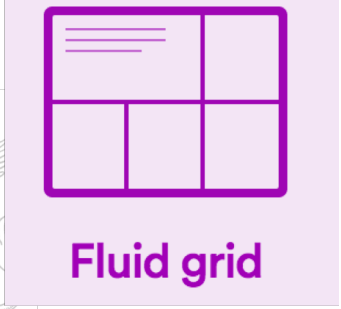


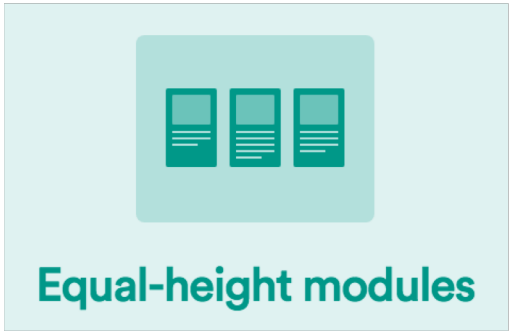


Megoldások gyakori problémákra









Példa

HTML

```
<html>
  <head>
    <meta charset="utf-8">
    <title>Flexbox example</title>
  </head>
  <body>

    <div id="flex-container">
      <div class="flex-item"></div>
    </div>

  </body>
</html>
```

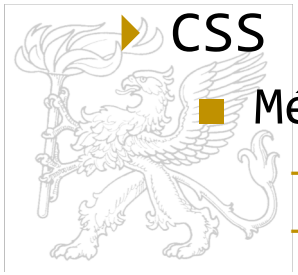
CSS

```
#flex-container {
  display: flex;
  border: 2px solid black;
  padding: 10px;
  height: 150px;
}

.flex-item {
  border-radius: 10px;
  background-color: purple;
  height: 50px;
  width: 50px;
}
```

Reszponzív web

- Úgy szeretnénk megírni a CSS-t, hogy sokfajta képernyőn is jól mutasson:
  - 27" asztali monitor
  - Macbook Air
  - Samsung Galaxy S7, iPhone 7, iPad
- Amennyiben nem jelezzük akkor a mobil böngészők **asztali képrenyő szélesség**-be jelenítik meg, ezt nagyítják ki
- Viewport-tal mondhatjuk meg a képernyő méretet
  - `<meta name="viewport" content="width=device-width, initial-scale=1">`
- CSS médium lekérdezés
  - Médium függő Szabályok
    - Relatív font méretek (em,rem)
    - Grid-View



```
@media (max-width: 500px) {
  article {
    margin: 0 2px;
  }
}
```

Összefoglaló

- Célok
- Követelmények
- Osztályzat
- Háttér
- Bevezető