بسم الله الرحمن الرحيم والصلاة السلام علي أشرف المرسلين سيدنا محمد ﷺ

How CSS Works Behind The Scenes

أزاي الـ CSS بتشتغل في الـ browser عشان تـ Display الـ web Pages بالشكل اللي بتشوفه داكدا

Table Of Contents

الخاتمة وشوية فضفضة

اذكر الله ❤

الزموا ذكر الله ففيه العون على متاعب الحياة ♥

اللَّهمَّ إنِّي أسألُكَ عِلمًا نافعًا، ورِزْقًا طيِّبًا، وعمَلًا مُتقبَّلًا.

اللَّهُمَّ أَعِنِّي عَلَى ذِكْرِكَ، وَشُكْرِكَ، وَحُسْنِ عِبَادَتِكَ.

اللهم صلّ وسلم وزد وبارك على نبينا محمد ﷺ

لا تنسونا من دعواتكم

By: Amr Elsayed

Three Pillars of Writing Good HTML and CSS (Never Forget Them!)

خلينا في البدية نتكلم عن اهم تلت مبادئ في الـ CSS واللي هما واللي لازم تكون علي وعي بيهم

1- Responsive Design

2- Maintainable and Scalable Code

3- Web Performance

Responsive Design

ودي ببساطة معناها أني أبني Web Page بشكل Responsive يعني قابل للعرض في جميع مختلف الشاشات يعني ع الـ Mobile يتعرض بشكل مناسب و علي الـ Laptops يتعرض بشكل مناسب وعلي الـ Large Screens يتعرض بشكل مناسب وهكذا ... تصميم الـ layout يتناسب مع حجم الشاشة اللي هيتعرض فيها

جميل جدا ... هل في حاجات لازم اراعيها عشان استخدم أو اعمل Responsive Design بشكل سليم ؟
قالك ايون فيه حاجات تراعيها لدا زي مثلا

1- Fluid Layouts
2- Media Queries
3- Responsive Images
4- Correct Units
5- Desktop First vs Mobile First Strategy

Maintainable and Scalable Code

ودي من أهم النقاط والمبادئ اللي لازم تكون عارفها ومعني النقطة دي انك تكتب Clean Code وحاجات تانيه كتير لازم تعمل حسابها .. تيجي نشوف .. تعالي نشوف

1- Clean Code
2- Easy To Understand
3- Growth
4- Reusable
5- How To Organize Files
6- How To Name Classes
7- How To Structure HTML

By: Amr Elsayed

Web Performance

في النقطة دي انت لازم تراعي ان الـ web App بتاعك يكون علي اعلي مستوي من الـ performance
وعشان يكون علي أعلي مستوي لازم يكون سريع ويكون صغير من حيث الحجم

في حاجات كتير جدا بتأثر علي الـ Performance

1- Less HTTP Requests
2- Less Code
3- Compress Code
4- Use a CSS Performance
5- Less Images
6- Compress Images

فكل ما تقلل حجم الصور وتستخدم preprocess زي الـ sass مثلا وتكتب كود قليل وفعال فدا يساعدك تعلي من الـ Performance بتاع الموقع

اذكر الله ❤

الزموا ذكر الله ففيه العون على متاعب الحياة ❤

اللهم صلّ وسلم وزد وبارك على نبينا محمد ﷺ

سبحان الله وبحمده عدد خلقه ورضا نفسه وزنة عرشه ومداد كلماته

لا إله الا انت سبحانك اني كنت من الظالمين

اللهم لك الحمد حمدًا طيبًا كثيرًا مباركًا فيه كما ينبغي لجلال وجهك ولعظيم سلطانك يا رب العالمين

لا حول ولا قوة الا بالله العلي العظيم

أَلا بِذِكْرِ اللَّهِ تَطْمَئِنُّ الْقُلُوبُ

لا تنسونا من دعواتكم

By: Amr Elsayed

How CSS Works Behind the Scenes: An Overview

اثناء تحميل صفحة الويب في حاجات كتير بتحصل خلف الكواليس زي http request و الـ right Domain و الـ name Services و حاجات تانيه كتير

بس احنا هنركز اكتر عن اللي بيحصل للـ Browser علي الـ Computer لما الـ user بيفتح الموقع

أول حاجة بمجرد ما الـ page تحمل بيحصل load للـ initial HTML File يعني ملف الـ HTML المبدأي كدا اول حاجة بتحمل
وبعد ما يتحمل بيحصل عمليه parse او تحليل سطر بسطر اثناء العملية دي ما بتحصل الـ Browser بيبني حاجة اسمها (DOM) Document Object Model واللي هي عبارة عن Web Document بيكون فيها family Tree للعناصر يعني div جنبه div تاني وجوا الاتنين Divs دول فيهم عناصر والعناصر دي جواها عناصر دي الـ Tree بتاعته

ببساطة بيتقال عليهم Parents واللي هما هنا الـ two divs و الـ Childrens واللي هما العناصر اللي جوا الـ Divs دي والـ Siblings الاشقاء يعني العناصر اللي جنب بعض اللي راسها براس بعض

بعد ما بيحصل parse للـ HTML File الـ Browser بيلاحظ وجود لسطر كدا مكتوب فيه تضمين او link بملف CSS

اول ما بيلاحظ دا بيعمل للملف دا load يعني بيـ load الـ css File وزي ما حصل للـ HTML File برضو الـ CSS File دا بيحصله Parse او تحليل هو كمان ولكن الـ parse دا بيكون مقعد بعض الشئ ودا اللي احنا عاملين الملخص دا عشان نناقشه اصلا

بس خليني اقولك ان اثناء عملية الـ parsing دي في حاجتين او خطوتين اساسيتين بيحصلو

أول واحدة وهي Conflicting CSS Declarations بيحصلها Resolving - حل ودا هنناقشه بالتفصيل قدام.

و خليني أقولك ان العملية دي ليها مسمي وهو الـ Cascade اول حرف في الـ Css

تاني حاجة بقا بتحصل اثناء عملية الـ parsing دي وهي Process Final CSS Values

يعني الـ margins بيتم قرأتها وتحويل قيمها لـ percentage units to px يعني اي برضو ؟!

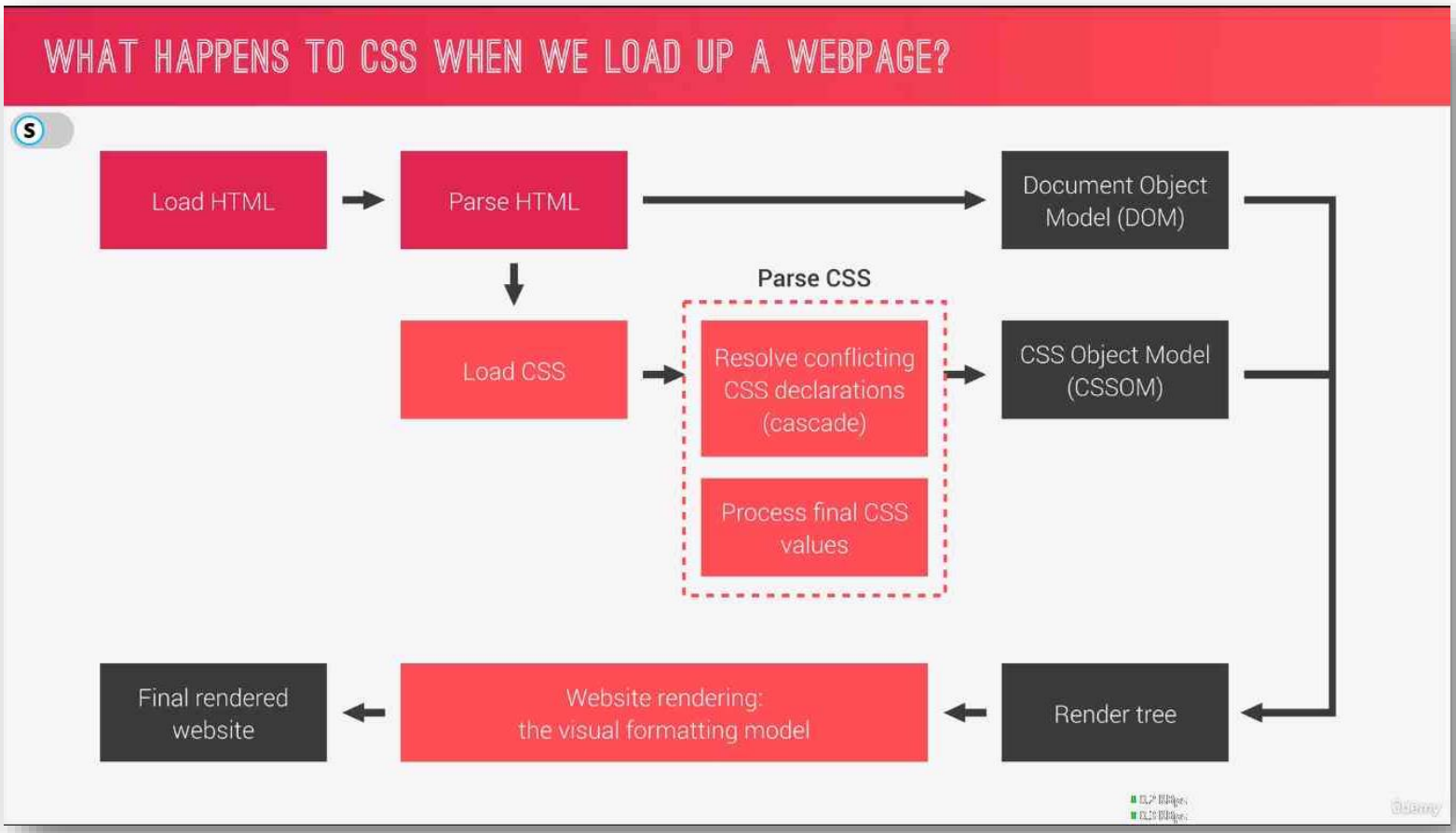
يعني تخيل معايا اني عامل margin-left: 50% مثلا ! بس الـ 50% دي مختلفه في الشاشات الصغيرة عن الشاشات الكبيرة في عملية التحليل دي بيتم ترجمتها او تحليلها للـ px unit لاستخدمها في مختلف الـ Devices

بعد ما الـ parsing دا بيحصل للـ CSS بيتم تخزينة في (CSSOM) CSS Object Model هي tree برضو زي الـ HTML كدا التخزين دا بيكون اسمه CSSOM

By: Amr Elsayed

بعد ما التحليل دا كله بيحصل بقا بيحصل عمليه اسمها Rendering لل Trees دي كلها مع بعض و عملية ال Render دي بتسمي بال formatting اللي ال browser بيستخدم فيها حاجة اسمها Visual Formatting Model وبرضو هنتكلم عنها قدام بالتفصيل اكتر

بعد ال Formatting دا ما يحصل بيحصل حاج اسمها Final Rendered Website وهنا انت بتقدر تشوف بقا الشكل بتاع الموقع بشكل فعلي علي المتصفح قدامك



اذكر الله ❤

الزموا ذكر الله ففيه العون على متاعب الحياة ❤

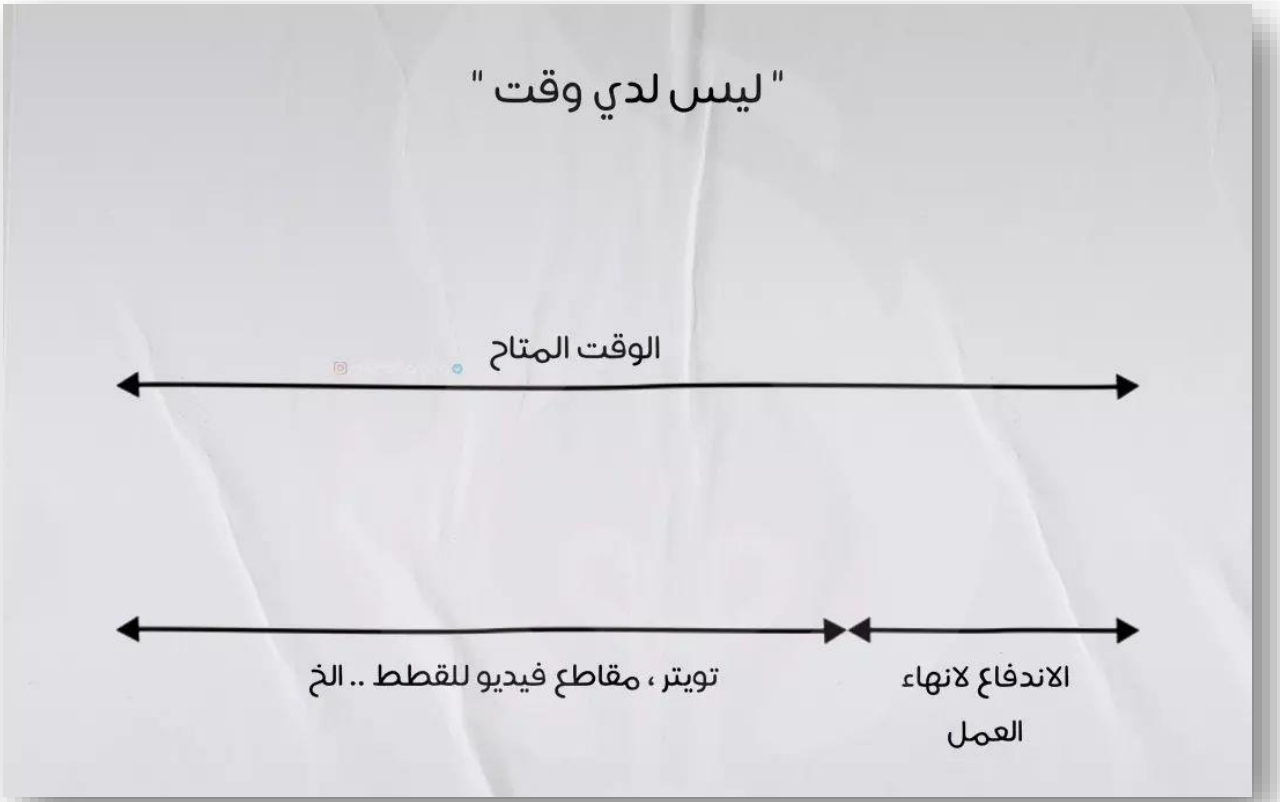
اللهم إني أستغفرك واتوب إليك، لا إله الا انت سبحانك إني كنت من الظالمين.

حسبي الله لا إله الا هو عليه توكلت وهو رب العرش العظيم.

أَلَا بِذِكْرِ اللَّهِ تَطْمَئِنُّ الْقُلُوبُ

By: Amr Elsayed

❤ راقت لي ❤



قبل ما تقول معنديش وقت شوف فعليا الوقت المتاح قدامك اي وشوف انت بتقضيه ازاي هتلاحظ ان كل وقتك او اهم فترات الوقت رايحة في الترفيه وجزء قليل جدا لأعمالك المهمة

اذكر الله ❤

سبحان الله وبحمده .. سبحان الله العظيم

أستغفر الله العظيم الذي لا اله الا هو الحي القيوم و أتوب إليه

حسبي الله لا اله الا هو عليه توكلت وهو رب العرش العظيم

لا تنسونا من دعواتكم

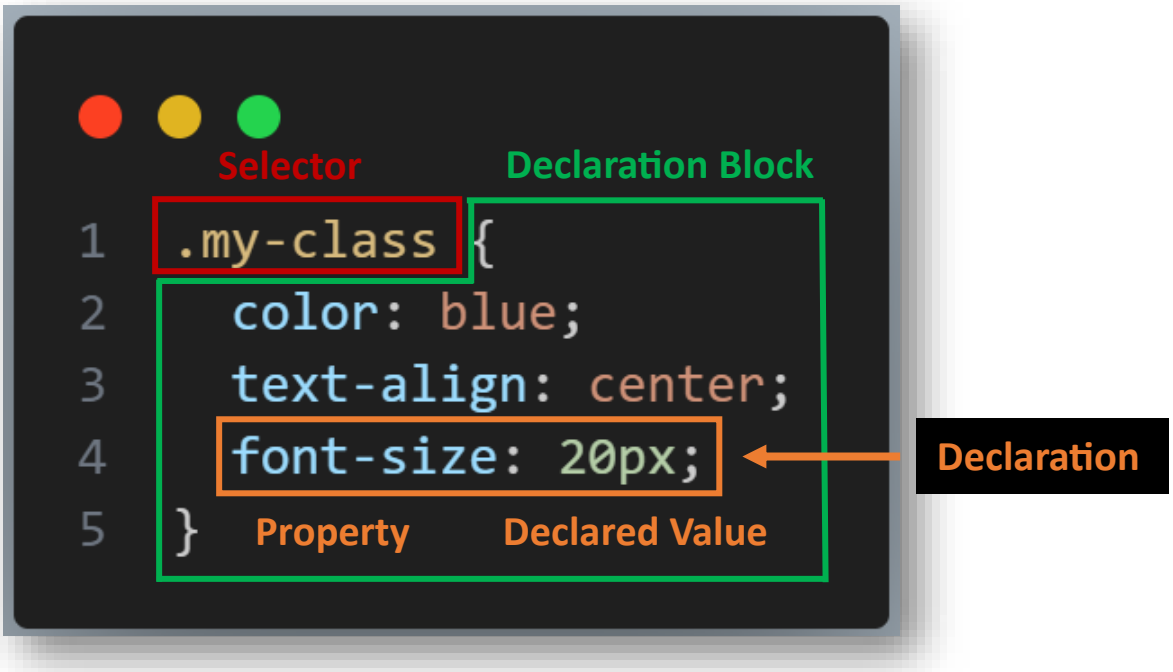
By: Amr Elsayed

How CSS is Parsed, Part 1: The Cascade and Specificity

في الدرس اللي فات خدنا overview عن ازي الـ HTML و الـ CSS بيحصلهم Process في المتصفح

في الدرس دا هنشوف عملية تحليل ملف الـ CSS بتتم ازاي

خلينا متفقين كلنا ان الـ Css بيكون ليه Rules معينة في طريقة الكتابة زي مثلا



والمقصود بالـ rules دي هو ان طريقة كتابة الـ css بتكون معتمدة علي وجود Selector اللي هو في حالتنا دي الـ .my-class واللي بكتبه عشان احدد اي عناصر الـ HTML اللي عندي اللي هديها التنسيقات اللي هتتكتب كمان معتمد علي وجود الـ Declaration Block واللي هي الخواص المكتوبة والقيم بتاعتها (تنسيقات العنصر)

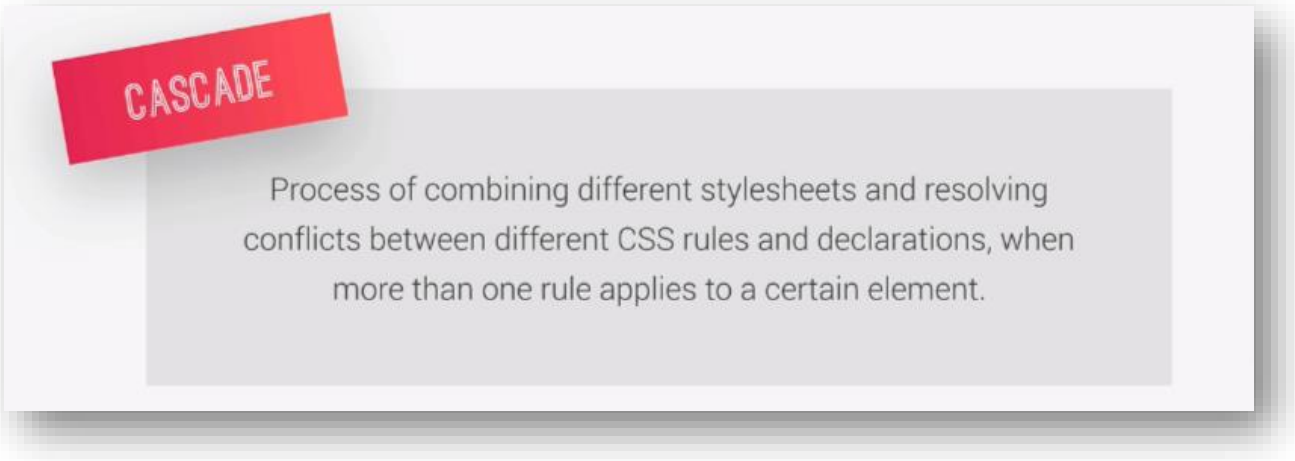
فزي ما انت شايف كدا

اول حاجة في عملية الـ parsing للـ CSS لو تفتكر لما قولت فوق وهي الـ Conflicting CSS Declarations

وتاني حاجة في عملية الـ Parsing للـ CSS لو تفتكر برضو قولت فوق وهي الـ Process Final CSS Values

By: Amr Elsayed

طيب خلينا نتكلم عن الـ Conflicting CSS Declarations واللي ليها مسمي وهو الـ Cascade



الـ cascade هي العملية اللي بيتم فيها دمج بين جميع ملفات الـ CSS المختلفة وحل الـ conflicts او التعارض الموجود بين كل الـ Declarations الموجودة في كل الملفات دي لو كانت متكررة لعنصر واحد

يعني ببساطة شديدة جدا لو انت عندك عنصر معين مسكته في الـ CSS و حطيت فيه بعض التنسيقات وروحت لملف تاني او في نفس الملف برضو تحت شوية مسكت نفس العنصر واديته نفس التنسيقات دي تاني الـ cascade بقا بيكون عامل زي مراقب كدا بيبص علي كل عنصر وتنسيقاته ولو متكررة بياخد منها نسخة واحدة
بس ويطبقها علي العنصر اللي تم اختياره دا

لنفترض انك مسكت class اسمه .my-class مثلا وحطيتله font-size: 20px
ونزلت تحت شوية مسكت نفس العنصر دا وحطيتله نفس الـ font-size: 20px او حتي غيرت قيمته

فاللي بيحصل هنا عملية اسمها Conflicting CSS Declarations او Cascade بتصلح الاختلافات اللي حصلت دي ويطبق أخر حاجة اتكتب

الـ CSS ممكون يجي من أكتر من مصدر .. طيب يعني اي بقا معني الكلام دا ؟ واي هي المصادر دي ؟!

اقولك انا

اول مصدر:

بيكون الـ Declarations اللي اتكتب من خلالك انت كـ Developer والمصدر دا بيتسمي الـ Author Declarations

تاني مصدر :

اللي بتيجي من الـ User علي سبيل المثال لما User معين يغير الـ Default font-size من الـ Browser

وتالت مصدر:

وهو الـ Default Browser Declaration واللي هي التنسيقات اللي بتتحط بشكل تلقائي من المتصفح نفسه

علي بعض العناصر والمصدر دا بيكون اسمه الـ user agent stylesheet
علي سبيل المثال الـ Anchor tag <a> مش بيكون له تنسيقات من المتصفح زي خط تحته ولون ازرق ؟!

بس فيه سؤال هنا ازاي فعليا الـ Cascade دا بيتم او بيحصل او بيعمل اي عشان يعالج الـ Conflicts دي كلها ؟!
ويتري في قواعد معينه بيمشي عليها عشان يحل الاختلافات دي ولا لا ؟!

اول حاجة بيبص عليها الـ cascade دا اول ما يجي يشتغل وهي الـ important flag!
بعد كدا الـ specificity بعد كدا الـ Source order (conflicts) اللي بتحصل منك انت كـ Developer
بالترتيب كدا

طيب تعالي نشوف الـ Important flag! دا عبارة عن اي

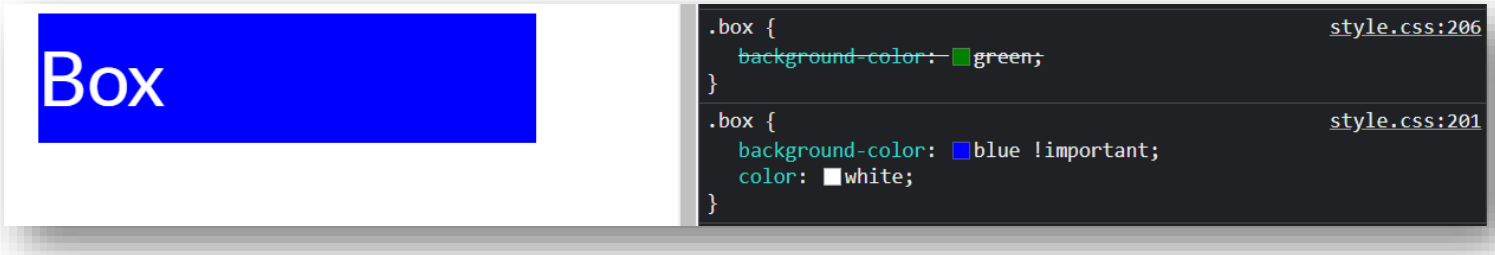
بص معايا الكود دا كدا

```html
<div class="box">Box</div>
```

```css
.box {
  background-color: blue !important;
  color: white;
}

.box {
  background-color: green;
}
```

يتري اي Style في دول اللي هيتطبق هل كلمة Box هيكون الـ Background بتاعها لونها أزرق ولا المفروض
هيكون لونها أخضر عشان انا اخر حاجة كتبتها للـ Box دا ;background-color: green
وبعدين مش المفروض أخر حاجة اكتبها للعنصر هي اللي تتطبق ؟!
تعالي نشوف



By: Amr Elsayed

اي اللي حصل ! دا طبق أول خاصية اللون الازرق مع ان الأخضر اتكتب تحته هنا دا اول حاجة بيبص عليها الـ Cascade وهو بيحلل ملف الـ CSS بيشوف العناصر وتنسيقتها ويبص ع الـ important! Flag ويطبقه

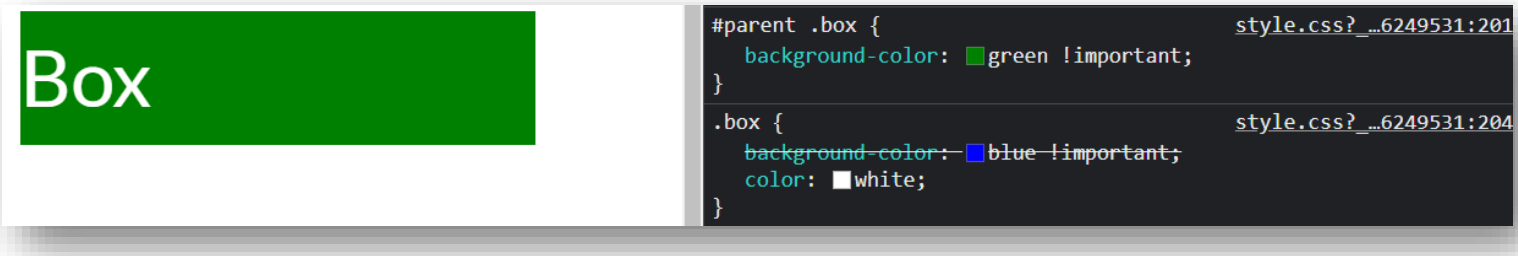
طيب لو الـ important! Flag دا متطبق في العنصرين هحدد ازي انهي خاصية اللي هتشتغل ؟!

وهنا يجي دور تاني مرحلة واللي هي الـ **specificity** أو الأقوي بص معايا ع الكود دا كدا

```html
<div id="parent">
  <div class="box">Box</div>
</div>
```

```css
#parent .box {
  background-color: green !important;
}
.box {
  background-color: blue !important;
  color: white;
}
```

دلوقتي انا عندي العنصرين بتوعي فيه Background-color والاتنين مطبق عليهم الـ important! Flag يتري مين فيهم اللي هيشتغل أكيد الاخير اللي هو الـ background-color: blue; لانه أخر حاجة اتكتب وهو اللي هيعمل overwrite علي اللي قبله !... تعالي نشوف



اي اللي حصل ؟! دا اللون الاخضر اللي فوق هو اللي اطبق ؟! ودا حصل بسبب specificity الاول أقوي عشان فيه ID و Class مع بعض لكن الـ box. بس لوحدها وهي Class كدا اضعف من الحالة الأولي لذلك هو طبق اللون الأخضر بسبب قوة الـ specificity مع ان اللون الأزرق جاي بعد الأخضر والمفروض هو اللي يـ overwrite عليه لكن دا محصلش بسبب الـ specificity زي ما قولنا

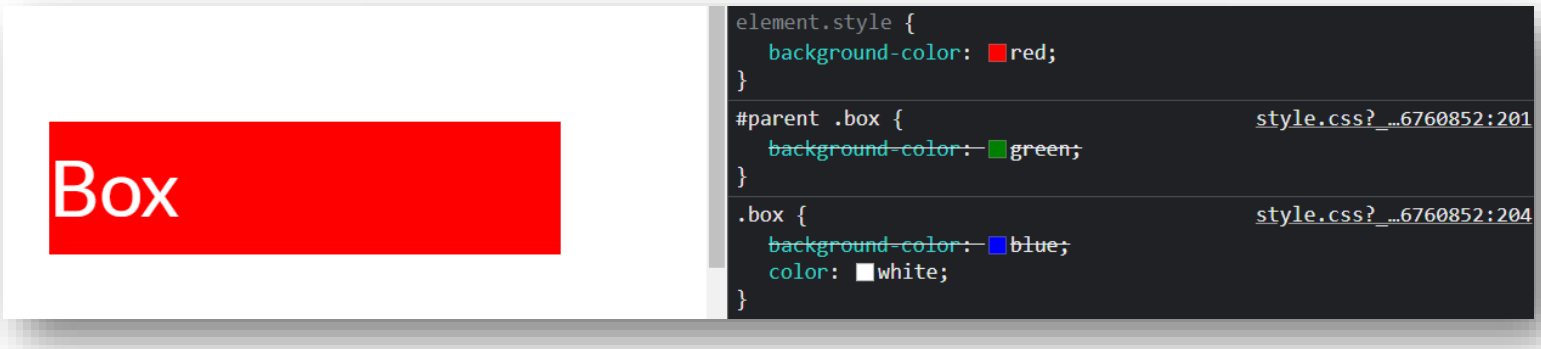
By: Amr Elsayed

ولاحظ ان الـ specificity هتكون أقوي حاجة لو الـ style معمول inline-style يعني انت حاطط التنسيقات في الـ HTML نفسه مع العنصر في حالة انك مش عامل !important لتنسيق معين يعني عندك

يعني انا قصدي التالي بص ع الكود دا عشان تعرف مين الـ specificity الأقوي

```html
<div id="parent">
  <div class="box" style="background-color: red">Box</div>
</div>
```

```css
#parent .box {
  background-color: green;
}
.box {
  background-color: blue;
  color: white;
}
```



الاقوي هنا هو الـ inline-style

By: Amr Elsayed

تعالي ناخد مثال اكثر تعقيدا ع الـ specificity

```html
<nav id="nav">
  <div class="pull-right">
    <a href="#" class="button">Don't Click Here</a>
  </div>
</nav>
```

```css
.button {
  font-size: 20px;
  color: white;
  background-color: blue;
}

nav#nav div.pull-right .button {
  background-color: green;
}

a {
  background-color: purple;
}

#nav a.button:hover {
  background-color: yellow;
}
```

ياتري انهو background-color في الاربعة دول اللي هتطبق ع العنصر بتاعي

الـ specificity بتتحسب بالترتيب كدا الاولية بتكون لل inline-style ثم الـ الـ ID ثم الـ class ثم الـ element

في العنصر الاول مفيش غير class واحد بس يبقي كدا المجموع (0 inline-style, 0 id, 1 class, 0 element)

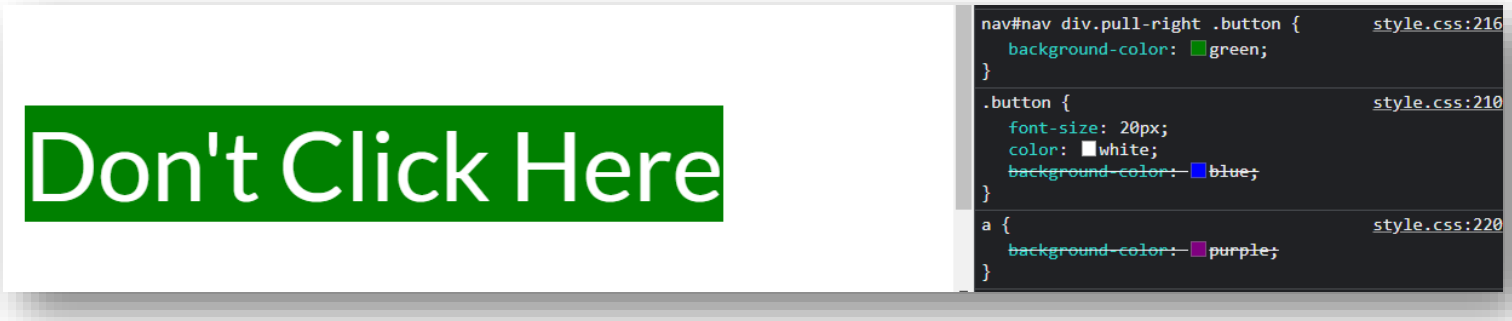
في العنصر التاني عندي الـ nav element و عندي #nav ID وعندي Element كمان اللي هو الـ div وعندي class اللي هو .pull-right وعندي class كمان اللي هو الـ .button
يبقي المجموع (0 inline-style, 1 id, 2 classes, 2 elements)

في العنصر التالت عندي element واحد بس اللي هو الـ a يبقي المجموع
(0 inline-style, 0 id, 0 class, 1 element)

By: Amr Elsayed

في العنصر الرابع عندي #nav id وعندي a element وعندي .button class. وعندي :hover class
يبقي المجموع (0 inline-style, 1 id, 2 classes, 1 element)

لو بصيت ع المجموع بتاع كل Selector هتلاحظ ان التاني هو أقوي واحد فيهم هو اللي الـ specificity بتاعته أعلي لذلك اللي هيطبق هو اللون الاخضر



طيب لو كل حاجة متساوية كل الـ Selectors نفس الـ specificity وكلهم نفس الـ !important Flag ومفيش اي inline-style هنا بقا يجي اخر حاجة بيبص عليها الـ Cascade و اللي هي الـ Source order

الـ Source order

ودي ببساطة الـ conflicts اللي بتحصل منك لو كل الـ Selectors زي بعض في كل حاجة هنا الـ Cascade هيشوف أخر حاجة اتكتب ويطبقها يعني أخر حاجة مكتوبه هتعمل overwrite علي الباقين

By: Amr Elsayed

يبقي ملخص سريع جدا ع اللي فات كله

اول حاجة بيبص عليها الـ cascade دا اول ما يجي يشتغل وهي الـ important flag!
بعد كدا الـ specificity بعد كدا الـ Source order (conflicts) اللي بتحصل منك انت كـ Developer

لو عندي عنصر في الـ HTML حطيت جواه inline-style باللون الأحمر
ومسكت نفس العنصر دا في الـ CSS اديته لون أخضر وحطيت معاه الـ important Flag! يبقي اللون الاخضر هو اللي هيتطبق لانه أقوي حاجة ودي أول حاجة بيبص عليها الـ Cascade ويطبقها ودي مرحلة الـ important Flag!

لو عندي عنصر في الـ HTML وملهوش اي inline-style وانا مش مستخدم الـ important Flag! بس في الـ CSS انا محدد العنصر دا بأكتر من طريقةالـ Cascade هيشوف انهو تحديده الأقوي فيهم ويطبقها علي العنصر وفوق في المثال الكبير عرفنا ازي بيحدد انهو Selector الأقوي ودي تاني مرحلة بعد الـ Important وهي الـ specificity

لو عندي عنصر في الـ HTML ليه inline-style او ملوش وانا مستخدم الـ important Flag! وكلهم زي بعض في الـ specificity الـ Cascade هياخد أخر حاجة اتكتب لل Selector دا ويطبقها علي العنصر ودي المرحلة اللي بيكون اسمها الـ Source order (conflicts)

يبقي الأولوية ديما بتروح لل important!

ثم الـ specificity واللي بيبص فيها لو فيه Inline-style لو مفيش بيبص علي الـ ID لانه بيكون تاني اقوي حاجة بعد الـ inline-style وبعدها بيبص ع الـ class ودا بيكون تالت أقوي حاجة وبعدين بيبص ع الـ element اللي بيكون رابع أقوي حاجة

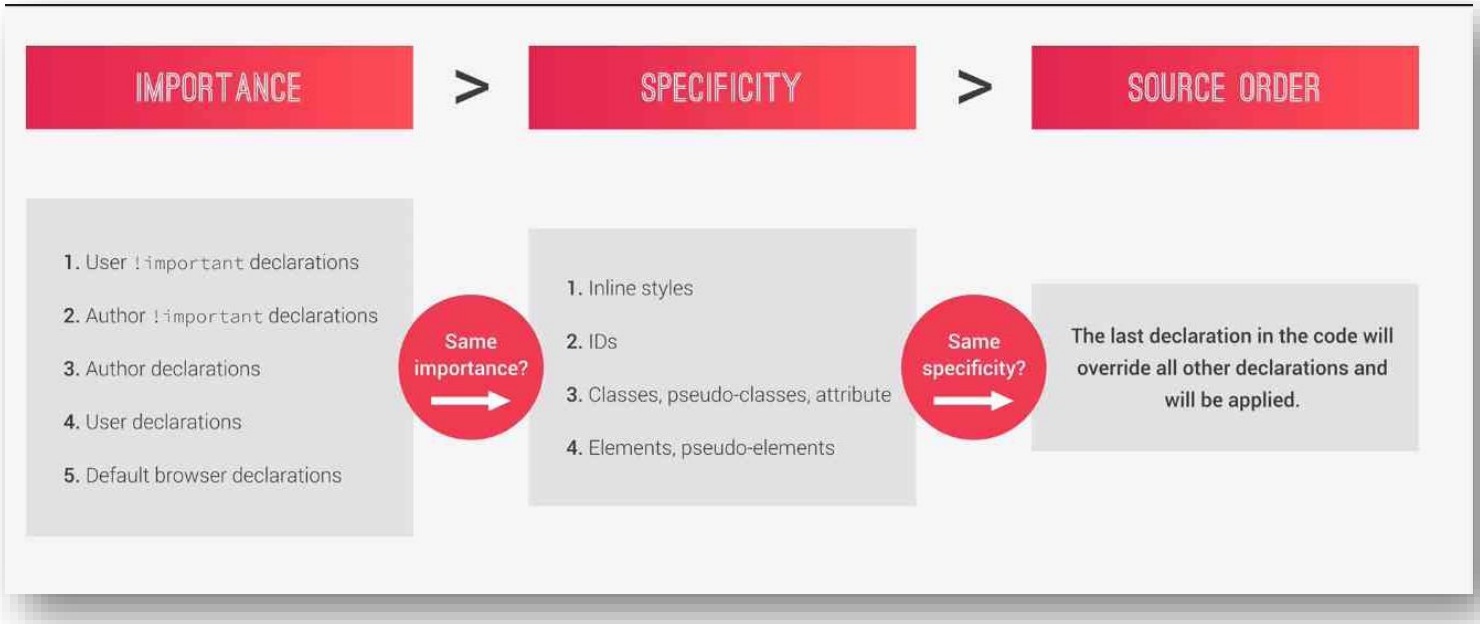
ثم لو كل حاجة زي بعض في كل حاجة نفس الـ important! ونفس الـ specificity يروح لأخر حاجة واللي هي الـ Source order (conflicts) بمعني انه هيشوف أخر حاجة اتكتب لل Selector ويطبقها ويكون ليها الأولوية في الظهور

ملحوظة مهمة جدا: خلي أستخدامك لل important! في أضيق الحدود الممكنة لو بس بتصلح كود بتاع شخص تاني وتنسيقاتك مش راضيه تتطبق لسبب ما استخدم الـ Important! لان زي ما قولنا بيكون اول واحد له الأولوية في التطبيق أن وجد

By: Amr Elsayed

Some Notes in English

- **CSS declarations marked with !important have the highest priority;**
- **But only use !important as a last resource. It's better to use correct specificities – more maintainable code!**
- **Inline styles will always have priority over styles in external stylesheets;**
- **A selector that contains 1 ID is more specific than one with 1000 classes;**
- **A selector that contains 1 class is more specific than one with 1000 elements;**
- **The universal selector * has no specificity value (0 ,0 ,0 ,0)**
- **Rely more on specificity than on the order of selectors;**
- **But, Rely on Order When using 3rd-party stylesheets – always put your author stylesheet last.**



❤ اذكر الله

❤ الزموا ذكر الله ففيه العون على متاعب الحياة

سبحان الله وبحمده سبحان الله العظيم

أَلا بِذِكْرِ اللَّهِ تَطْمَئِنُّ الْقُلُوبُ

By: Amr Elsayed

Specificity in Practice

تعالي ناخد تمرين علي الـ Specificity عشان نعمق فهمه اكتر ونوضح بعض حاجات برضو

شوف الكود دا معايا

```html
<nav id="nav">
  <div class="pull-right">
    <a href="#" class="button">Don't Click Here</a>
  </div>
</nav>
```

```css
.button {
  font-size: 20px;
  color: white;
  background-color: blue;
}

a {
  background-color: purple;
}

#nav div.pull-right a.button {
  background-color: orangered;
}


#nav a.button:hover {
  background-color: yellow;
}
```

مين الأقوي هنا واللي هيطبق الـ Declarations اللي فيه ؟!

تعالي نحسبها تاني بحسبة بسطية جدا

By: Amr Elsayed

تخيل ان عندي اربع حاجات هما اطراف المقارنة وهما (inline-style, ID, Class, Element)

يبقي الاربع حاجات اللي فوق دا هبص علي كل Block عندي فوق واشوف الـ Selector بيحتوي علي ايه من الاربع حاجات دي

قبل ما ابدأ المقارنة انا معنديش اي inline-style لذلك هو ديما هيكون بصفر 0 في المقارنة دي

اول Selector

عندي اللي هو الـ .button عبارة عن class واحد يبقي المقارنة كالتالي :

(0 inline-style, 0 id, 1 class, 0 element)

تاني Selector

عندي اللي هو الـ a عبارة عن Element واحد يبقي المقارنة كالتالي :

(0 inline-style, 0 id, 0 class, 1 element)

تالت Selector

عندي فيه التالي

1- #nav ادي كدا أول ID
2- div Element أدي كدا أول Element
3- .pull-right أدي كدا أول Class
4- a Element ادي كدا تاني Element
5- .button أدي كدا تاني Class

يبقي المقارنة كالتالي :

(0 inline-style, 1 id, 2 class, 2 element)

يبقي بشكل مبدأي كدا اللون الـ orangered في تالت Selector هو اللي هيطبق

حتي لو نقلت الـ Selector دا او الـ Block دا كدا كله علي بعضه فوق هو برضو اللي هطبق وقولنا ليه ؟!
عشان الـ Specificity بتاعته أقوي واحد فيهم

By: Amr Elsayed

طيب ليه لما جيت أعمل Hover تأثير الـ Hover مظهرش ؟! واتحول لـ اللون الاصفر ؟!

عشان برضو هنا تأثير الـ Specificity في الـ block بتاع الـ Hover أضعف من اللي اللي فوق
هتقولي ازاي ؟!
هقولك تعالي نعمل مقارنة تاني

تالت Selector عندي فيه التالي

1. #nav ادي كدا أول ID
2. div Element أدي كدا أول Element
3. .pull-right أدي كدا أول Class
4. a Element ادي كدا تاني Element
5. .button أدي كدا تاني Class

يبقي المقارنة كالتالي :

(0 inline-style, 1 id, 2 class, 2 element)

رابع Selector عندي فيه

1- #nav ادي كدا أول ID
2- a Element ادي كدا أول Element
3- .button أدي كدا أول Class
4- :hover أدي كدا تاني Class

يبقي المقارنة كالتالي :

(0 inline-style, 1 id, 2 class, 1 element)

وبالمناسبة الـ Pseudo Class تصنف كـ Class وبيتم احتسابها في مقارنة الـ Specificity

كدا مازال تالت Selector أقوي حتي الـ Hover مش هيتم تطبيقه غير لو بقا مساوي لتالت Selector في القوة

By: Amr Elsayed

```css
#nav div.pull-right a.button {
  background-color: orangered;
}

#nav div.pull-right a.button:hover {
  background-color: yellow;
  color: black;
}
```

كدا كافة المقارنة رجحت لحساب رابع Selector بالتالي تأثير الـ Hover هيتم تطبيقه

لاحظ انك ممكن لو استخدمت الـ !Important هيكون ليه الأولوية في التنفيذ والتطبيق لأنه زي ما قولنا فوق في الدرس اللي قبل دا أول حاجة فوق خالص ان الـ Cascade أول حاجة بيبص عليها أو بيديها الأولوية هي الـ !important Flag لو تفتكر انا برضو هجبلك اسكرين من الجزء دا عشان تفتكر

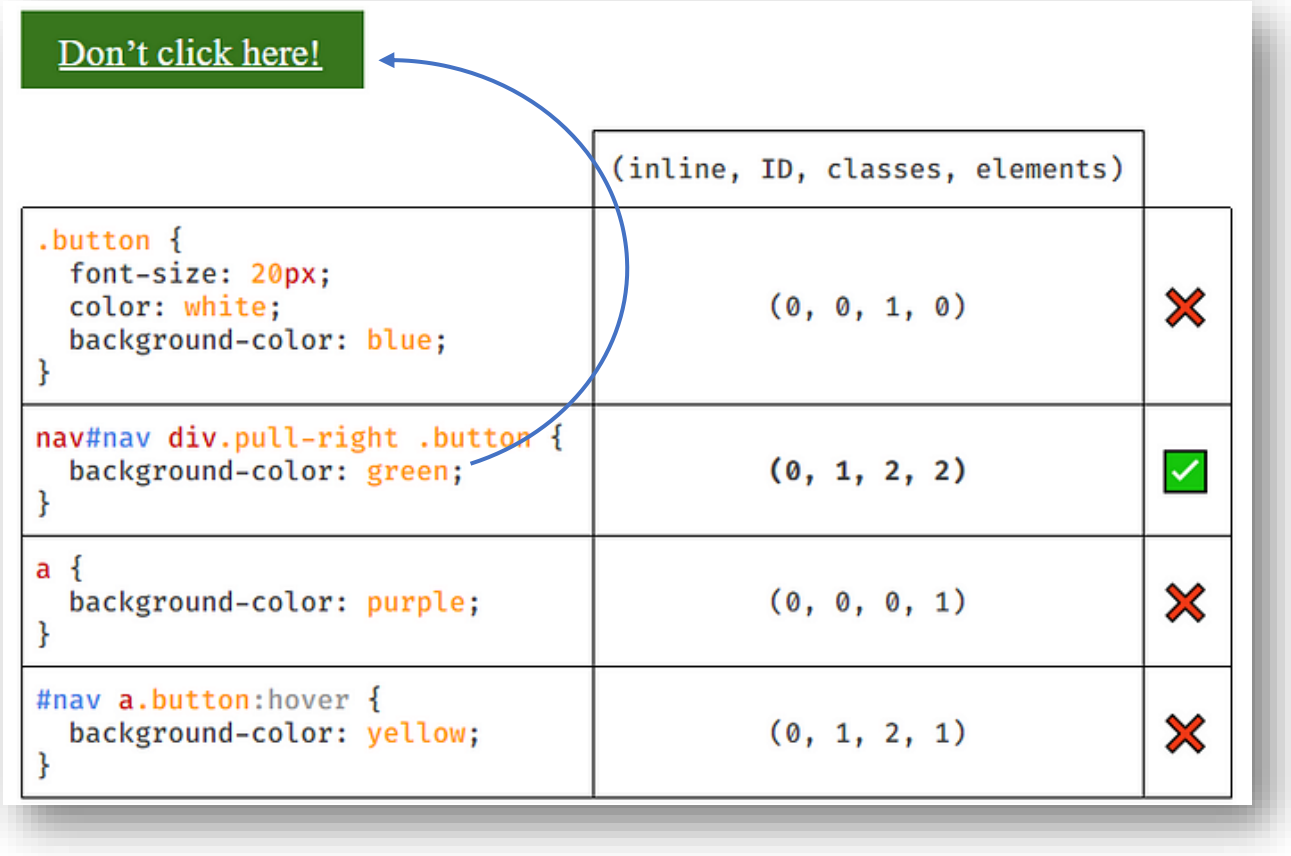
بس انا واي حد عموما مش بيفضل أستخدام الـ !important Flag غير في ظروف طارئة جدا والسبب في دا بيرجع لان الكود بتاعك يكون maintainable أو قابل للصيانة ديما مع استخدامك لل !important Flag بيكون

الوضع صعب شوية خصوصا لو التطبيق كبير جدا وعندك اكتر من عنصر ليهم نفس الـ Specificity

فبلاش استخدام الـ !important Flag كتير خصوصا لو انت كـ Developer اللي بتكتب الـ Code بايدك

By: Amr Elsayed

صورة مختصرة للمقارنة بشكل توضيحي أفضل فيها بعض الاختلافات عن الكود اللي فوق بس الـ Concept يعني



اذكر الله ❤

الزموا ذكر الله ففيه العون على متاعب الحياة ♥

لا إله الا الله وحده لا شريك له .. له الملك وله الحمد وهو على كل شيء قدير.

استغفروا الله العظيم الذي لا إله الا هو الحي القيوم واتوب اليه.

اللهم صلّ وسلم وزد وبارك على نبينا محمد ﷺ.

لا إله الا انت سبحانك إني كنت من الظالمين.

أَلا بِذِكْرِ اللَّهِ تَطْمَئِنُّ الْقُلُوبُ

أفتكرونا بدعوة طيبة

By: Amr Elsayed

How CSS is Parsed, Part 2: Value Processing

في الدرس دا هنفهم ازي الـ Values بتاعت الـ Properties بيحصلها تحليل في الـ CSS كمان هنعرف ازاي الـ Percentage % و الـ CSS Units زي و الـ em و الـ rem و الـ vh بترجم او بتتحسب ازاي في الـ CSS

طيب في سؤال حلو انا سامعه من الاستاذ اللي واقف هناك دا بيقولي ليه انا محتاج اعرف الـ units دي بتتحسب ازاي ؟!

عشان في كل مرة بتستخدم فيها unit غير الـ px وليكن بتستخدم الـ rem و الـ em تبقي فاهم ان الـ units دي بتتحول لـ px برضو في النهاية بس السؤال هنا بقا المهم جدا هو ازاي بتتحسب وازاي بتتحول لـ px ودا اللي هنناقشه في الدرس دا .. يلا بينا

بص معايا كدا علي الأكواد دي وفي أخر الدرس انا هعملك جدول لامم كل اللي هيتشرح لحد ما قبل الجدول دا

```html
<div class="section">
  <p class="amazing">CSS is absolutely amazing</p>
</div>
```

```css
.section {
  font-size: 1.5rem;
  width: 280px;
  background-color: orangered;
}
p {
  width: 140px;
  background-color: green;
}
.amazing {
  width: 66%;
}
```

By: Amr Elsayed

بالنظر كدا لل CSS هلاقي انه عندي Declaration 6 اللي هما font-size و width و background-color و width تاني و background-color تاني و width تالت مرة

تعالي نشوف بقا ازاي بيحصلهم analysis في ال CSS

تعالي نبدأ بال p ونشوف ال width اللي فيه

بشكل مبدأي كدا انا عندي هنا conflict حاصل بسبب اني عملت Select لل p وعملت Select لل amazing.
اللي هو اصلا Class ال P في ال HTML

انهو width اللي هيطبق ؟! لو قولت ال amazing. عشان هو أخر حاجة ال css هتشوفها وتترجمها هقولك صح
بس لو انا نقلت ال class دا فوق ال p هل هتكون دي اجابتك برضو ؟! اعتقد لا
لان في سبب تاني لو ركزت شوية هتلاحظ هنا ان ال specificity بتاعت ال class أقوي من ال element لو تفتكر

يبقي ال width: 66% هو اللي هيطبق علي ال P مين اللي حدد دا هل انا او انت ؟! لا ال Cascade بال Rules بتاعته
حدد مين الأقوي ومين ليه الأولوية في التطبيق

طيب ال 66% دي زي ما اتفقنا هتتحول لل px ازاي دا بيتم بقا .. دا بيتم ع اكتر من مرحلة او في مراحل في النص كدا بتحصل زي اي

1- Declared Value (Author Declarations)
2- Cascaded Value (After The Cascade)
3- Specified Value (Defaulting Value If There Is No Cascaded Value)
4- Computed Value (Converting Relative Values To Absolute)
5- Used Value (Final Calculations, Based On Layout)
6- Actual Value (Browser And Device Restrictions)

يبقي دي كل ال process اللي ال CSS بتمشي عليها عشان تحول أي unit لل pixel في النهاية وطبعا هنفصصهم واحدة واحدة

أول مرحلة: ال Declared Value ودي المرحلة اللي بيكون فيها القيمة اللي انت ك author او ك Developer كتبتها وفي حالتنا دي هي ال 66% في ال amazing class. و ال 140px في ال p

تاني مرحلة: ال Cascaded Value ودي المرحلة اللي ال CSS بتحل فيها ال Conflicts واول ما تشوف Conflict حصل بين ال amazing. و ال p بتشوف مين الأقوي وتختارة وبناءا عليه ال 66% هتتطبق

لحد كدا انت معايا وسليم وتمام وكله زي الفل ؟! فاهم صح ؟ تمام

By: Amr Elsayed

تالت مرحلة: الـ Specified Value

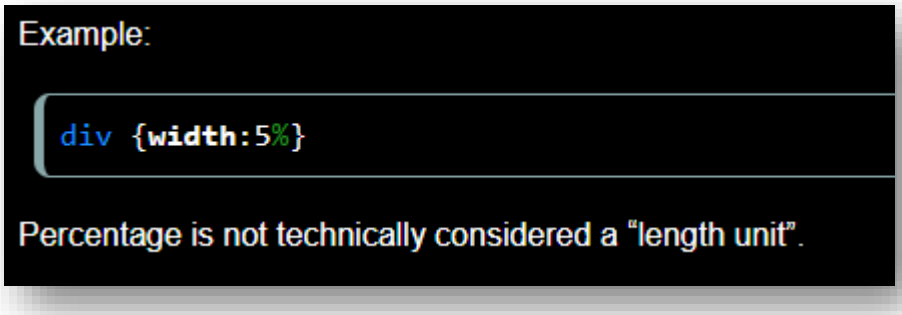
في المرحلة دي لو مفيش اي Declared Value بيتحط حاجة اسمها initial Value وهنعرف ازاي دي بتيجي او بتيجي منين لما نخلص الخاصية دي بمراحلها الـ 6 ونشوف خاصية تانية

في حالتنا وبما ان انا حطيت value كـ Developer يبقي الـ 66% هتتطبق مدام فيه Cascaded Value هيبقي هتطبق وهتبقي الـ Specified Value بـ 66% هي كمان

رابع مرحلة: الـ Computed Value

في المرحلة دي الـ values اللي بتكون Relative او نسبية ببتحول لـ px بحيث يتم توريثها وهنتكلم بالتفصيل عن التوريث او الـ inheritance .. كمان في المرحلة دي الـ CSS keywords زي bolder, orange واللي زيها بيحصل Compute وبتتبدل في المرحلة دي

بس عشان الـ % تكنيكالي مش unit ايون زي ما بقولك كدا ففي الحالة دي هتفضل 66% زي ما هي برضو وهثبتلك حالا ان الـ % مش unit فعلية



لينك

لينك أخر

خامس مرحلة: الـ Used Value

ودي بقا المرحلة اللي بيحصل فيها تحويل فعلي من الـ % لـ px في حالتنا احنا عندنا الـ .section الأب في الكود فوق كان فيه width بالقيمة 280px لذلك الـ .amazing هي 66% من الـ 280px دول يعني بتساوي 184.8px

هي استمدت قيمتها من الـ Parent Element اللي هو الـ .section الأب

سادس مرحلة: الـ Actual Value

في المرحلة دي الـ CSS بتكون محددة جدا او specific بتقرب القيمة دي لـ 185px

يبقي الـ 185px هي القيمة الفعلية اللي هتتحط في النهاية

By: Amr Elsayed

تعالي نشوف خاصية تانيه ونشوف الـ value بتاعتها بيحصلها parse ازاي

وليكون تعالي نشوف الـ padding في المثال بتاعنا هتقولي استني بس انا مكتبتش padding اصلا
هقولك صحيح بس انا عايز تعرف ان في خواص انت مش بتكتبها بيتحطلها قيم معينة فعليا بشكل default
فتعالي نعدي الـ padding اللي مش موجود دا اصلا علي الـ 6 مراحل اللي عندي فوق

السبب في الكلام دا ان كل element في الصفحة بيكونله كل CSS property محتاجه يكون ليها Value حتي لو انت معملتهاش Declaration

فتعالي نشوف الـ padding اللي احنا متكبتنهوش دا مع الـ 6 مراحل بتوعنا

أول مرحلة: الـ Declared Value مفيش خالص اي Declared Value

تاني مرحلة: الـ Cascaded Value مفيش خالص اي Cascaded Value لان مفيش declaration حصل

تالت مرحلة: الـ Specified Value اااه في هنا بقا Specified Value او الـ default
ودا بسبب ان كل واي CSS property بيكون ليها initial value وببساطة دي بتكون الـ Value الفعلية
لو مفيش Cascaded Value ودا اللي انا قولتلك هنجيلها كمان شوية فوق لما كنت بتكلم في الخاصية اللي
فاتت ففي الحالة دي (الـ initial Value هتكون بـــ 0px) وبيني وبينك التوريث بيكون ليه عامل هنا بس
هنتعرف علي دا بالتفصيل قدام

رابع مرحلة: الـ Computed Value بشكل فعلي الـ 0px هي absolute unit يعني هو 0px والـ unit بتاعته px فعليا
لذلك المرحلة دي واللي بعدها هتفضل 0px او خلاص مش محتاجينهم

خامس مرحلة: الـ Used Value هتكون 0px

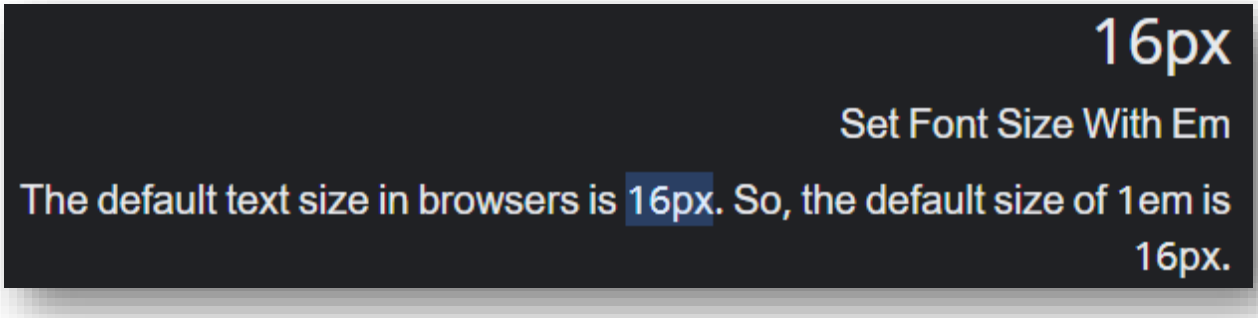
سادس مرحلة: الـ Actual Value هتكون 0px

تعالي نشوف الـ font-size اللي موجود في الـ .section واللي هو 1.5rem بس قبل ما نشوف دي تعالي نناقش حاجة تانيه وهنرجع للـ font-size اللي موجود في الـ .section

احنا محتاجين نعرف الـ font-size بتاع الـ root بتاعنا بتكون 16px بشكل تلقائي زي ما انا وانت والدنيا كلها عارفه

ان الـ Browsers بتحط Default Font-Size بالقيمة 16px

يبقي الـ root الـ font-size بتاعه بيكون 16px ودا برضو هيعدي ع الـ 6 مراحل بتوعنا



أول مرحلة: الـ Declared Value مفيش خالص اي Declared Value لانها 16px من الـ browser نفسه

تاني مرحلة: الـ Cascaded Value هنا الحالة دي مختلفة عن الـ padding هنا فيه Cascaded Value طيب هتقولي ازاي ؟!
هقولك فاكر لما قولنا ان الـ CSS بتيجي من أكتر من مصدر ؟ من ضمن المصادر دي اللي هو الـ user agent stylesheet واللي هي هنا فرضت الـ font-size للـ page بـ 16px كـ default Value

تالت مرحلة: الـ Specified Value بتتحط كـ initial value لو مفيش Cascaded value وبما ان فيه Cascaded Value جاية من الـ user agent stylesheet يبقي الـ Specified Value هتكون بـ 16px هي كمان

رابع مرحلة: الـ Computed Value هتكون 16px

خامس مرحلة: الـ Used Value هتكون 16px

سادس مرحلة: الـ Actual Value هتكون 16px

By: Amr Elsayed

تعالي بقا نشوف الـ font-size اللي موجود في الـ .section واللي هو 1.5rem بعدنا ما عرفنا الـ font-size لل root

```
.section {
  font-size: 1.5rem;
```

يلا نعديه ع الـ 6 مراحل

أول مرحلة: الـ Declared Value واللي هي 1.5rem و الـ rem هي Relative unit لذلك هتتحول لل px

تاني مرحلة: الـ Cascaded Value واللي هي هيتم قرأتها كـ 1.5rem برضو

تالت مرحلة: الـ Specified Value هتكون 1.5rem مدام فيه Cascaded Value يبقي الـ Specified value هتكون زي الـ Cascaded Value

رابع مرحلة: الـ Computed Value هتكون 24px ... ازاااي بقا .. تعالي اقولك مش انت سألتني ازاي سيبني أجاوب بقا سعتك :"D

الـ rem unit بتكون نسبية او Relative لل root font-size يعني هي بتستمد قيمتها من الـ root وبما ان الـ root له Font-size بشكل default بـ 16px

يبقي الـ .section الـ Font-size بتاعه هيكون 1.5 * 16px بتساوي 24px

خامس مرحلة: الـ Used Value هتكون 24px

سادس مرحلة: الـ Actual Value هتكون 24px

تعالي نشوف الـ font-size اللي موجود في الـ p هتقولي بس انا محطتش font-size للـ p Element ولا حتي للـ class اللي في الـ p اللي هو الـ amazing. هقولك انا ثانية واحدة بس انت سألت نفسك الأول ازاي الـ p الـ text اللي فيه ليه font-size او ظاهر اصلا مع انك فعليا مكتبتش اي Font-size ؟!
صح مش كدا ولا اي ؟!

في الحالة دي بقا دا لا هو فيه default value محطوطة للـ p عشان يظهر كدا ولا فيه اي initial Value هتقولي الله !! انا كدا احتارت اومال الـ p دا واخد Font-size وظاهر كدا منين وليه هو الـ font-size بتاعه اللي محطوطله مخليه شكله كدا؟! يعني ليه هو مش اصغر من كدا وليه مش أكبر من كدا ليه هو شكله كدا تحديدا ؟!

دا حصل بسبب ميكانزيم تاني اسمه الـ inheritance او التوريث الـ p دا يا صديقي ورث حجم الخط اللي هو فيه دا من الأب بتاعه اللي الـ section. اللي هو قيمته بـ 1.5rem اللي فعليا بتساوي 24px

اكبر دليل علي كلامي روح علي الـ p دا حطله font-size بالقيمة 24px من عندك مش هتلاقي اي تغير حصل ليه؟ عشان هو ورث حجم الخط دا من الأب بتاعه اللي هو الـ section.

طيب ليه بيحصل الميكانزيم دا ؟! لان ببساطة تخيل انك عندك اكتر من p جوا الأب بتاعهم انك تديهم كلهم font-size حاجة مش عملية خالص ورخمة لذلك هي بترث من الاب اللي هي فيه وهنتكلم عن التوريث بالتفصيل وازاي بتم برضو متقلقش

ما تيجي نبص ع الـ 6 مراحل بتوعنا

أول مرحلة: الـ Declared Value مفيش اي Declared Value

تاني مرحلة: الـ Cascaded Value مفيش اي Declared value عشان يحصلها Cascade بالتالي مفيش اي Cascaded Value

تالت مرحلة: الـ Specified Value هتكون 24px بطريقة الـ inheritance من الأب

رابع مرحلة: الـ Computed Value نفس الـ Value بـ 24px

خامس مرحلة: الـ Used Value نفس الـ Value بـ 24px

سادس مرحلة: الـ Actual Value نفس الـ Value بـ 24px

By: Amr Elsayed

تعالي بقا أعملك الجدول اللي هلملك فيه كل الكلام اللي فات دا

	width (paragraph)	padding (paragraph)	font-size (root)	font-size (section)	font-size (paragraph)
1- Declared Value (Author Declarations)	140px 66%	-	-	1.5rem	-
2- Cascaded Value (After The Cascade)	66%	-	16px (Browser default)	1.5rem	-
3- Specified Value (Defaulting Value If There Is No Cascaded Value)	66%	0px (initial value)	16px	1.5rem	24px
4- Computed Value (Converting Relative Values To Absolute)	66%	0px	16px	24px (1.5 * 16px)	24px
5- Used Value (Final Calculations, Based On Layout)	184.8px	0px	16px	24px	24px
6- Actual Value (Browser And Device Restrictions)	185px	0px	16px	24px	24px

INHERTANCE

و خليني الخصلك كل اللي فات دا في كلمتين كدا هنا برضو

- كل Property بيكون ليها initial value بيتم اللجوء اليها لو مفيش اي Declared values او مفيش اي inheritance
- الـ Browsers بتخصص Default font-size لـ Page بالقيمة 16px
- النسبة المئوية % وكل الـ الواحدات النسبية او الـ Relative Units بيتم تحويلها لـ pixels

أفتكرونا بدعوة طيبة

By: Amr Elsayed

❤ عبارات تحفيزية ❤

لو أن الناس كلما استصعبوا أمرا تركوه ماقام للناس دنيا ولادين

-عمر بن عبدالعزيز

اذكر الله ❤

الزموا ذكر الله ففيه العون على متاعب الحياة ❤

سبحان الله و الحمد لله و لا اله الا الله و الله أكبر و لا حول ولا قوة الا بالله

اللهم صلّ وسلم وزد وبارك على نبينا محمد ﷺ

اللهم إني أستغفرك واتوب إليك، لا إله الا انت سبحانك إني كنت من الظالمين

أَلا بِذِكْرِ اللَّهِ تَطْمَئِنُّ الْقُلُوبُ

By: Amr Elsayed

How Units Are Converted From Relative To Absolute (px) - Value Processing

تعالي بقا نفهم بالتفصيييل الشديد ازاي الـ Engine بتاع الـ CSS بيحول الـ Relative Units زي الـ rems و الـ ems والـ vw والـ vh للـ Pixels في المرحلة الرابعة و الخامسة (الـ Computed Value) و (الـ Used value)

بص معايا ع الكود دا كدا

```
1 html, body {
2   font-size: 16px;
3   width: 80vw;
4 }
5
6 header {
7   font-size: 150%;
8   padding: 2em;
9   margin-bottom: 10rem;
10  height: 90vh;
11  width: 1000px;
12}
13
14.header-child {
15  font-size: 3em;
16  padding: 10%;
17}
```

بشكل مبدأي عايز أقولك انه فيه فرق بين استخدام النسب المئوية % للخطوط و استخدامها لقياسات الطول والعرض والمسافات وهنفهم دا حالا بالتفصيل الشديد

وخليني افكرك ان الـ % مش unit بشكل تيكنيكال هي مش unit بس عايزك تعتبرها كدا المرادي عشان الشرح

By: Amr Elsayed

اول حاجة خليني امسك الـ header انا فيه عملت declaration لـ font-size بـ 150%
ودا معناه انه الـ header هيبص ع الأب بتاعه اللي هو الـ body و الـ html هلاقي الـ font-size بـ 16px

فدا معناه ان 150% * 16px بيساوي 24px

يبقي الـ header الـ font-size بتاعه هيكون 24px

ولو تفتكر من خمس ست سطور قولتلك انه فيه فرق بين استخدام النسب المئوية % للخطوط fonts
و استخدامها لقياسات الطول والعرض .... ففي الـ header انا استخدمت النسبة المئوية مع الـ fonts

تعالي نشوف الفرق بقا اثناء استخدمها للطول والعرض

ملحوظة: لما بنستخدم الـ % مع الـ length زي الـ height والـ width والـ padding و الـ margin
بيكون التعامل هنا مختلف عن استخدام الـ % مع الـ fonts الفرق هنا بيكون كالتالي

الـ length لما بتستخدم فيه الـ % وهي بتحسب الـ computed Value بيكون المرجع بتاعها الـ width بتاع الـ parent - الاب اللي الـ Element دا نفسه موجود فيه

اكيد انت مش فاهم حاجة وانا هشرحلك دلوقتي بالتفصيل

تعالي نشوف الـ padding اللي موجود في الـ .header-child اللي هو بالقيمة 10%

```
.header-child {
  padding: 10%;
```

هيحسب قيمة الـ padding دا من الـ width بتاع الـ Parent بتاعه اللي هو الـ 1000px

```
header {
 width: 1000px;
```

فدا معناه ان 10% * 1000px بيساوي 100px

ومرة تانية بأكد فيه فرق بين استخدامك لـ % و الـ units كلها عموما في الـ fonts زي الـ font-size
وبين استخدامك ليهم في الـ length زي الـ padding و الـ margin والـ height و هكذا هتفهم اكتر قدام

By: Amr Elsayed

الـ ems والـ rems هما font-based units يعني ديما بتبص ع حجم الـ font-size هو دا مرجعها

اي الفرق بين الـ em و الـ rem
الـ em بيكون الـ reference بتاعها الـ element نفسه في بعض الحالات او الـ parent بتاع الـ element دا
الـ rem بيكون الـ reference بتاعها الـ root font-size

عندي في الكود اللي اتكتب فوق فيه em موجودة في الـ font-size بتاع الـ .header-child

```
.header-child {
  font-size: 3em;
```

الـ css هتحسبها ازاي ؟! اتفقنا ان الـ em بيكون المرجع بتاعها يا العنصر نفسه يا الاب اللي هو فيه
وبما ان الاب اللي العنصر دا فيه الـ font-size بتاعه 150% اي 24px يبقي الـ 3em هتكون كالتالي
3em * 24px بيساوي 72px يبقي الـ .header-child الـ font-size بتاعه بـ 72px

فيه em موجودة في الـ padding بتاع الـ header

```
header {
  padding: 2em;
  font-size: 150%;
```

الـ padding دا length هنا بتكون فيه حسابات تانيه عن الـ fonts هنا الـ padding بـ 2em هيكون المرجع بتاعه هو الـ font-size بتاع الـ element دا نفسه هنا المرجع بتاعها هيكون العنصر نفسه تحديدا الـ font-size بتاع العنصر

وبما ان العنصر دا فيه الـ font-size بـ 150% اي 24px يبقي الـ 2em هتكون كالتالي
2em * 24px بيساوي 48px

هتقولي طيب افرض الـ header دا مكنش فيه الـ font-size بـ 150% اللي هي 24px كان الـ 2em دي هتتحسب ازاي ؟!

هقولك يا معلم ركز الـ em بيعتمد ع الـ font-size بس في حالتين الحالة الأولي لو فيه font-size في الـ element نفسه يبقي هو هيعتمد عليه لو مفيش بيروح يعتمد ع الـ font-size بتاع الـ parent بتاعه

لو بقا الـ header مفهوش font-size كان الـ 2em دي هتروح تشوف الـ root او الـ Body الـ font-size فيه بكام وتحسب علي اساسه

By: Amr Elsayed

وفيه rem موجودة في الـ margin-bottom بتاع الـ header

```
.header {
  margin-bottom: 10rem;
```

زي ما قولنا الـ rem بيكون الـ reference بتاعه الـ font-size بتاع الـ root في الحالة دي هو 16px
اذًا 10rem * 16px بيساوي 160px

الـ ems والـ rems هما font-based units يعني ديما بتبص ع حجم الـ font-size هو دا مرجعها

اي الفرق بين الـ em و الـ rem
الـ em بيكون الـ reference بتاعها الـ element نفسه في بعض الحالات او الـ parent بتاع الـ element دا

فاكر لما قولتلك السطرين اللي فوق دول انا هنا كان قصدي اي ؟! تعالي اديك مثال عشان تفهم قصدي
لنفترض ان هغير الـ padding اللي في الـ .header-child دا من 10% لـ 2em

```
.header-child {
  font-size: 3em;
  padding: 2em;
}
```

الـ padding هنا هيتحسب ازاي ؟!
انا قولتلك اي .. قولتلك حاجتين الـ ems بتعتمد ع الـ font-size وقولتلك ان المرجع بتاعها بيكون يا الـ element نفسه في حالة وجود font-size فيه ... يالا parent .. هنا مدام فيه font-size في الـ element اللي هي فيه هتبص علي الـ font-size بتاع الـ element نفسه
يبقي الـ padding هنا هيكون كالتالي
الـ font-size بتاع الـ element هو 3em و الـ 3em بتاعته دي في الحالة دي اعتمدت ع الـ parent اللي هو 150% اي 24px فالـ Font-size هنا كان 3em * 24px بـ 72px

وبما ان الـ font-size بتاع الـ element بـ 72px يبقي الـ padding بالـ em هيكون المرجع بتاعها الـ font-size دا
يبقي الـ 2em * 72px بيساوي 144px

عارف انا لو شيلت الـ font-size اللي في الـ element دا الـ padding هيتسحب ازاي ؟! فكر ثواني كدا
الـ padding بالقيمة 2em هيكون المرجع بتاعه الـ font-size بتاع الـ parent اللي هو 150% اي 24px
فهتكون الحسبة كالتالي 2em * 24px بتساوي 48px .... انت فاهم صح اكيد فاهم مفيهاش نقاش

By: Amr Elsayed

انت دلوقتي ممكن تكون بتسأل نفسك سؤال مهم جدا ليه استخدام الـ rems و الـ ems وهي معتمدة علي الـ Font-Size ؟! ... عشان في الـ layouts المعقدة والكبيرة بمجرد ما تغير الـ font-size لعنصر ما الـ padding والـ margin والـ length بتاعه هيتغير بشكل تلقائي لانه معتمد ع الـ font-size ودي بيخلي التصميم بتاعك فيه flexibility كبيرة جدا وجامد جدا جدا ومبني كله علي بعضه واي تغير بتعمله بيخلي التصميم مرن

واخيرا عندنا اخر اتنين Relative units اللي هما الـ vh و الـ vw

بشكل مبدأي الـ 1vh بيساوي 1% من الـ viewport height بتاع الشاشة

```
.header {
  height: 90vh;
```

فدا معناه ان الـ 90vh دي هي بمثابة 90% من الـ current viewport height

90vh * 1% بيساوي 90%

والـ 1vw بيساوي 1% من الـ viewport width بتاع الشاشة

```
html, body {
  width: 80vw;
```

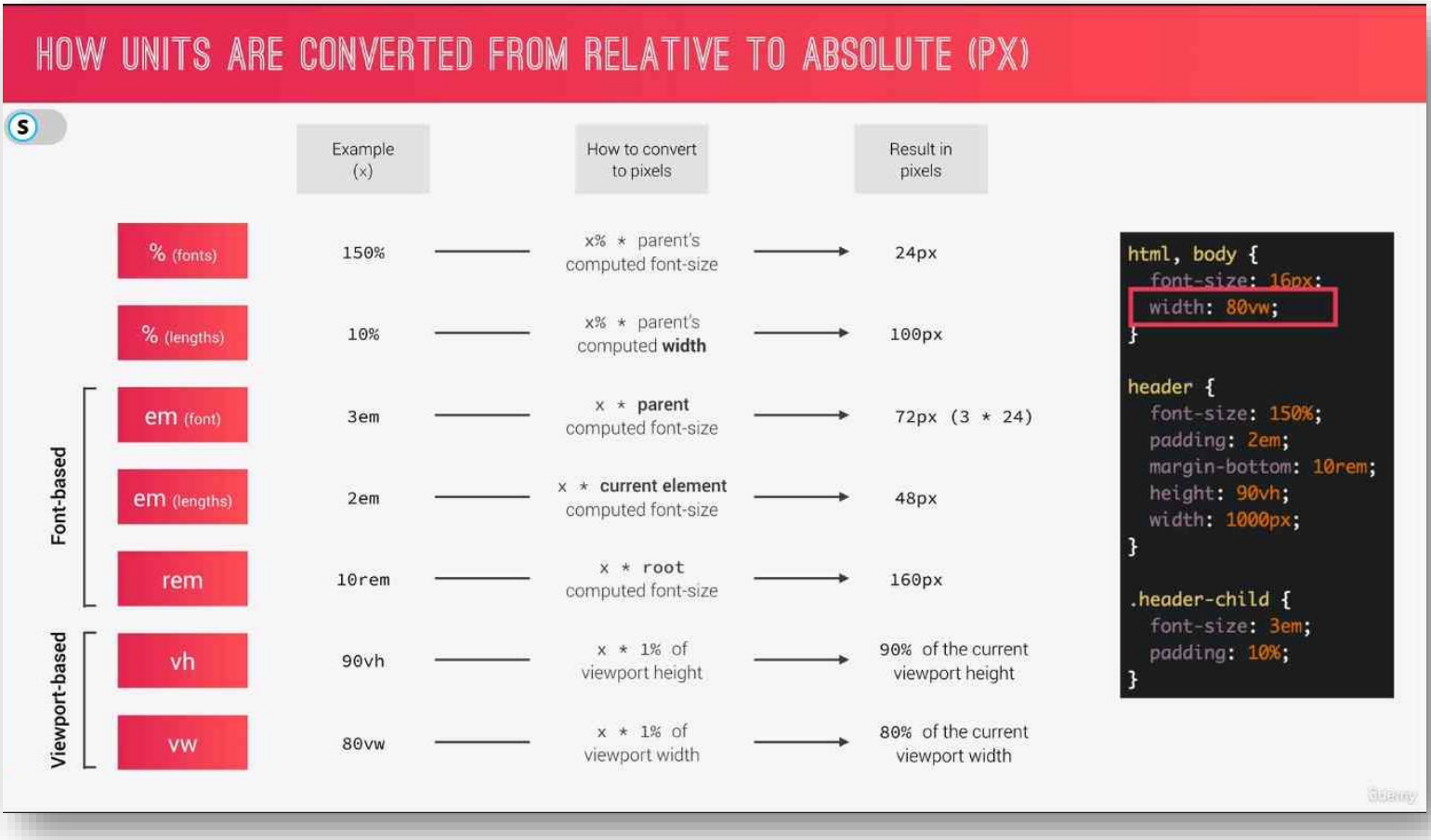
فدا معناه ان الـ 80vw دي هي بمثابة 80% من الـ current viewport width

80vh * 1% بيساوي 80%

الـ viewport دا بيتحدد من الـ browser نفسه

By: Amr Elsayed

دي صورة لاممكن فيها الدنيا كلها بمجرد ما تقراها هتفهم كل اللي فات



و خليني الخصلك كل اللي فات دا في كلمتين كدا هنا برضو

- كل Property بيكون ليها initial value بيتم اللجوء اليها لو مفيش اي Declared values او مفيش اي inheritance
- الـ Browsers بتخصص Default font-size للـ Page بالقيمة 16px
- النسبة المئوية % وكل الـ الواحدات النسبية او الـ Relative Units بيتم تحويلها لـ pixels
- النسبة المئوية % في الـ child لو بتستخدمها في الـ font-size بيكون قياسها relative لـ Font-size بتاع الأب
- النسبة المئوية % بتكون Relative لـ Width بتاعع الأب لو بتستخدمها مع اي lengths زي الـ padding
- الـ em بيتم قياسها نسبةً لـ font-size بتاع الأب لو بتستخدمها في خاصية الـ font-size لـ child
- الـ em بتيم قياسها نسبةً لـ current font-size بتاع الـ element نفسه لو بتستخدمها مع الـ lengths
- الـ rem ديما وابدا بتكون Relative او بيتم قياسها نسبةً لـ font-size بتاع الـ root element
- الـ vh و الـ vw هما ببساطة الـ view port's height and width respectively يعني بيتم تحديدهم بناءًا علي الـ viewport بتاع الشاشة نفسها والمتصفح اللي بيحددها

By: Amr Elsayed

How CSS is Parsed, Part 3: Inheritance

اتكلمنا عن الـ inheritance في الدروس اللي فاتت وجيه الوقت عشان نشوف الـ css بتتعامل ازاي مع الـ inherit

الـ inheritance أو التوريث هي طريقة بيتم فيها توريث بعض الخصائص والقيم من الأباء للأبناء

شوف معايا الكود دا كدا وتعالي نشوف الـ line-height الـ child. هيورثه ازاي من الـ parent.

```css
.parent {
  font-size: 20px;
  line-height: 150%;
}

.child {
  font-size: 25px;
}
```

في كذا سيناريو محتاجين ندرسهم الأول

كل CSS Property لازم يكون ليها Value

السيناريو الأول وهو ان الـ CSS بتسأل سؤال هل فيه Cascaded Value ؟!

لو أه => فيه يبقي الـ specified Value هتكون هي الـ Cascaded Value

وخلي بالك الـ الـ specified Value اللي هي بتكون الـ init value لو مفيش Cascaded Value

والـ Cascaded Value بتتوجد لما يكون فيه Declared Value منك انت كـ Developer

لو لأ => هيحصل سيناريوهات تانية

السيناريو الأول: لو مفيش Cascaded Value .... هل الـ Property دي inherited ؟!

لو أه => يبقي الـ specified Value هتبقي الـ Computed Value بتاعت الـ Parent

لو لأ => يبقي الـ specified Value هتتحط بالـ init value

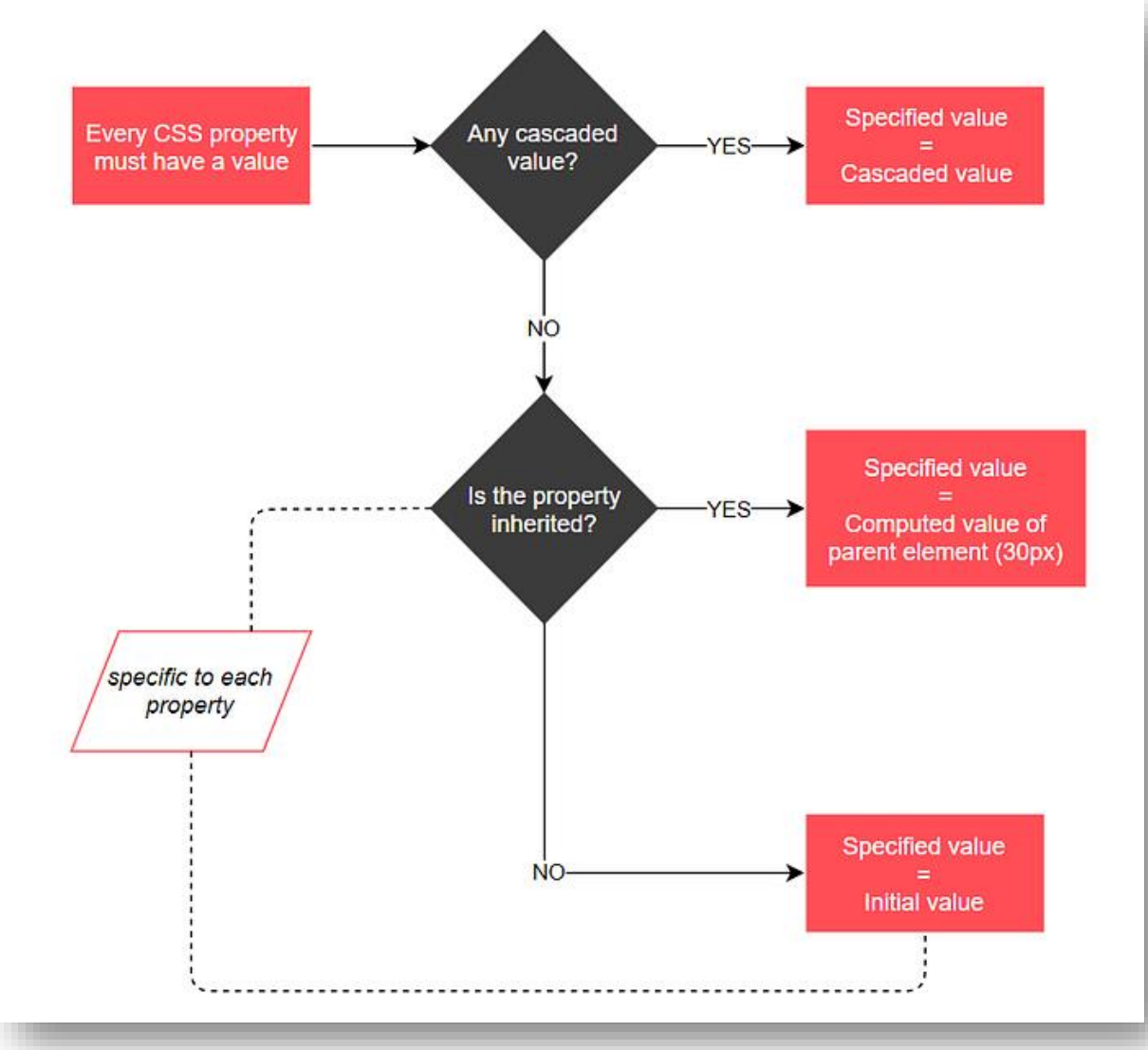
By: Amr Elsayed

طيب في المثال بتاعنا في الكود اللي فوق
الـ CSS هتسأل فيه Cascaded Value هنا للـ line-height في الـ child. ؟!
لا مفيش
يبقي هنسأل سؤال تاني هو الـ line-height دا خاصية من الخواص اللي ممكن تتورث عادي ؟!
لو الاجابة أه وهي أه فعلا
يبقي الـ child. دا هيكون الـ line-height بتاعه بـ 30px

ودا حصل من خلال أن الـ line-height بتاع الاب اللي بـ 150% لما يتحول للـ px
هيتعمد علي الـ Font-size بتاع الاب اللي هو 20px
فيبقي المعادلة كتالي 150% * 20px بتساوي 30px

يبقي الـ line-height بتاع الـ parent. بـ 30px
والـ line-height من الخواص اللي بيتم توريثها عادي والـ child. هيورثها يبقي الـ line-height بتاع الـ child.
هيكون هو كمان بـ 30px

مخطط توضيحي للفكرة اللي نقشناها فوق



By: Amr Elsayed

ملحوظات مهمة:

- الـ child. بيكون فيه كل خواص الـ css عادي بس بالـ init value بتاعتها وفي المثال بتاعنا الـ child. موجود فيه الـ line-height بالـ init value بـ 0 بس عشان الـ line-height بقا من الخواص اللي بتتورث عادي من الاباء للأبناء فـ الـ line-height هيتم توريثته لـ child. فهنا الـ init value اللي بـ 0 هتتشال ويتحط مكانها نفس القيمة اللي في الـ parent. اللي هي 30px وحسبناها فوق

- في خواص كتير مبيتمش توريثها زي الـ padding مثلا وخصائص تانيه كتير وتقدر تشوف الخواص اللي بيتم توريثتها والخصائص اللي لا

- الـ inheritance هو امداد الابناء ببعض الخواص من الأباء لو الخواص دي قابلة انها تتورث ودا بشكل أو بأخر بيخليك تكتب كود اقل وبيحقق شرط من الشروط اللي اتكلمنا عليها في أول صفحة من الملخص وهي ان الكود بتاعك يكون more maintainable code

- كل الـ properties اللي متعلقة بالـ text بتكون inherited زي الـ font-size والـ color و الـ font-family وكدا وكل الخواص زي الـ padding والـ margins مش بيتم توريثها لانه فقط تخيل انك عندك Section مديله padding بـ 60px فتخيل لو كل عنصر جو الـ section دا ورث الـ padding دا !! شئ فظيع جدا طبعا :"D

- خلي بالك ان الـ computed value اللي القيمة بتاعتها بالـ px في شكلها النهائي هي اللي بيتم توريثها وليس الـ declared values

- الـ inheritance بيشتغل في الابناء لو مفيش Declared value للخاصية اللي تم توريثها دي يعني لو انت كتبت في الـ parent خاصية بيتم توريثها وروحت للابن كتبت نفس الخاصية بقيم تانيه هنا الـ inheritance مش هيشتغل

- ممكن تستخدم الـ inherit keyword عشان تجبر اي خاصية مكتوبة في الابن انها تورث من الأب

- ممكن تستخدم الـ initial keyword عشان تعمل reset لأي property لـ init value بتاعتها

أفتكرونا بدعوة طيبة

By: Amr Elsayed

❤️ راقت لي ❤️

اذكر الله ❤️

الزموا ذكر الله ففيه العون على متاعب الحياة ❤️

لا إله الا الله وحده لا شريك له .. له الملك و له الحمد وهو على كل شيء قدير

أَلَا بِذِكْرِ اللَّهِ تَطْمَئِنُّ الْقُلُوبُ

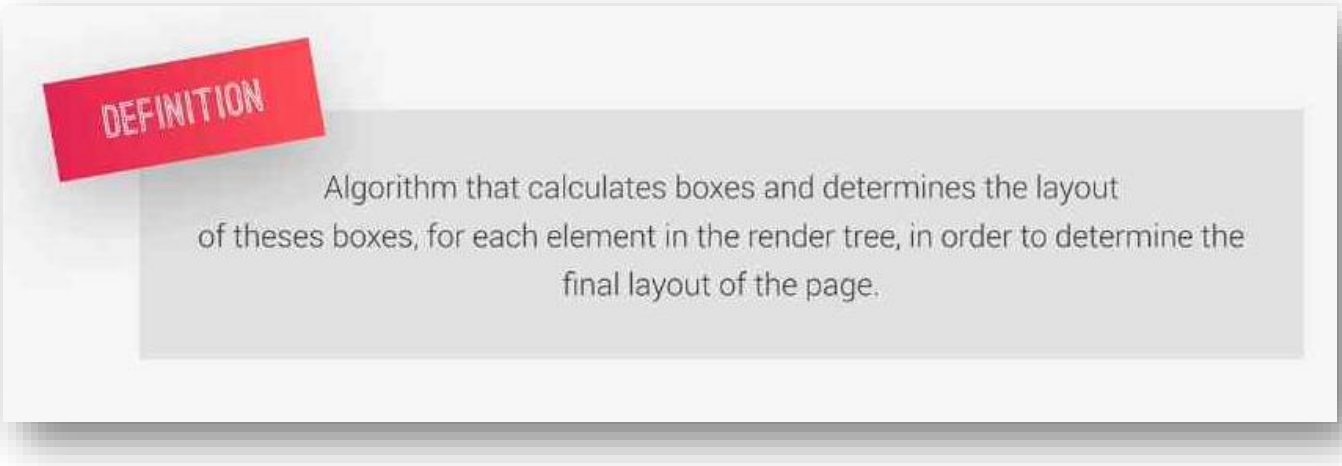
لا تنسونا من دعواتكم

By: Amr Elsayed

How CSS Renders a Website: The Visual Formatting Model

بعد ما اتكلمنا عن الـ Parsing Phase او مرحلة تحليل الـ CSS للـ code
دلوقتي نتكلم فيه عن الـ Website Render Phase أو مرحلة ظهرو الـ Website بشكله المرحلة دي
بيتم استخدام فيه ألية عمل اسمها Visual Formatting Model

ماهو الـ Visual Formatting Model



هو ببساطة algorithm بيستخدم لحساب الـ Boxes و تحديد الـ layout لكل الـ boxes دي لكل عنصر بيحصله
render بالاضافة لتحديد الـ final layout of the page يعني هو اللي بيحسب قياسات الـ box-model والـ Floats و الـ flex و الـ positions والـ layout بشكل عام

In summary, the Visual Formatting Model provides the foundation for rendering and positioning elements on a web page, enabling developers to create visually appealing and well-structured web layouts.

The layout of these boxes is governed by:

- **Box dimensions (the box model)**
- **Box type (inline, block, inline-block)**
- **Positioning scheme (normal flow, float, and absolute positioning).**
- **Stacking-context**
- **Relationships between elements in the document tree.**
- **External information (e.g., viewport size, intrinsic dimensions of images, etc.).**

يبقي الـ Visual Formatting Model بوظيفة هو حساب كل النقاط اللي اتذكرت فوق دي وهنتعرف عليهم واحد واحد دلوقتي

By: Amr Elsayed

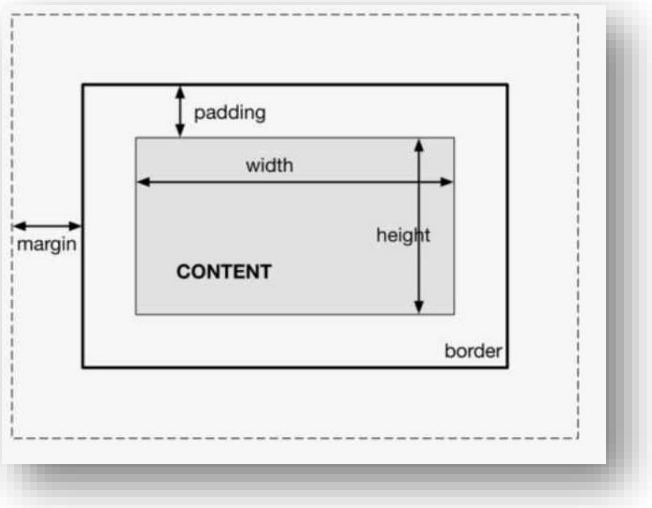
Box dimensions (the box-model)

الـ Box-model هو واحد من أهم مبادئ لغة الـ CSS وهو حاجة لازم تتقنها لتصميم صفحة ويب.

الـ Box-model هو أحد العوامل اللي بتحدد ازاي بيتم عرض العناصر على صفحة الويب وكيفية تحديد حجمها.

وفقًا لـ Box-model، يمكن اعتبار كل عنصر على صفحة الويب عبارة عن مربع مستطيل.

ويمكن لكل مربع أن يحتوي على width و height و padding و margins و border



ومع ذلك، لاحظ أن كلهم ممكن يتوجدو بشكل اختياري، عشان ممكن يكون فيه عناصر او boxes بدون margins أو padding او border

ومن خلال فهم واستخدام الـ Box-model ، يمكنك إنشاء layouts متجاوبة وجذابة من الناحية البصرية. عشان بيمكنك من التحكم في حجم العناصر وتحديد مواقعها وتباعدها، وتنفيذ layouts مرنة بتتكيف مع مقاسات الشاشات المختلفة.

أزاي الـ Box-model بيحسب الـ width و الـ height بتاع العنصر

total width = right border + right padding + specified width + left padding + left border

total height = top border + top padding + specified height + bottom padding + bottom border

Example: calculate height

=

0px top border + 20px top padding + 100px specified height +
20px bottom padding + 0px bottom border

=

140px

By: Amr Elsayed

مش ملاحظ حاجة اننا لو حددنا width معين أو height معين زي الـ 100px اللي في المثال بنلاحظ ان الـ padding و الـ border بيزودو من حجم الـ height دا وبيتحسبوا معاه ؟! بس دا بشكل ما مش حاجة كويسة

عشان كدا الحل اللي بنسخدمه عشان نمنع الموضوع دا هو خاصية الـ box-sizing بالقيمة border-box لانك لما تحدد height معين وليكن 100px وتزود عليه padding-top و padding-bottom بـ 20px لكل واحد

لما بتستخدم الـ box-sizing هو بخلي العنصر دا دا الـ height بتاعه ككل بالـ 100px لانه بيخلي الـ height الفعلي بـ 60px وبيحسب الـ padding بـ 40px فيكون المجموع الكلي للعنصر دا 100px فقط

Example: calculate height after using box-sizing

=

0px top border + ~~20px top padding~~ + 100px specified height +

~~20px bottom padding~~ + 0px bottom border

=

100px

By: Amr Elsayed

Box type (inline, block, inline-block)

الـ type او نوع الـ box بيتحدد من خلال خاصية الـ display وكل عنصر الـ html بيكون عليه default display زي الـ div بيكون block وزي الـ span بيكون inline وهكذا

وطبعا انت مدام بتقرا الملخص دا فانت شخص فاهم CSS كويس وبتزود معلوماتك مفيش داعي اشرحلك الفرق بينهم انا هكتفي بصورة فقط فيها مقارنة بينهم ابقي أقراها يعني

Block

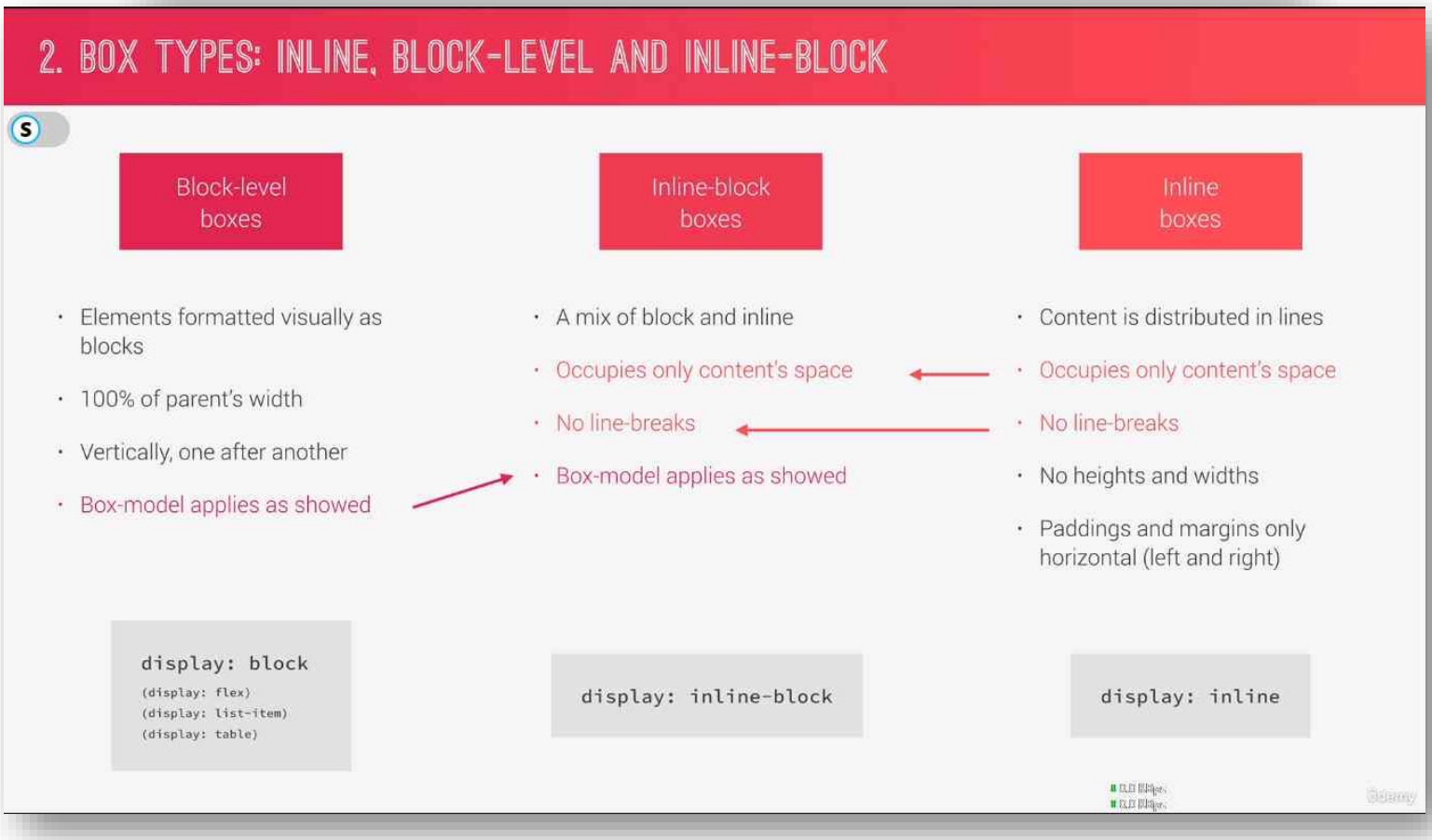
- بياخد الـ full-width لو لم يتم تحديد width
- بيعمل خط فاصل بينه وبين العناصر اللي هتيجي تحته عشان يمنعها تيجي جنبه
- بيحترم الـ padding - و الـ margin - و الـ Width - و الـ height بمعني انه بينفذ أي أمر ليهم

Inline

- لا يحترم الـ Width و الـ Height
- لا يضيف خط فاصل بينه وبين العنصر اللي تحته .. يعني أي عنصر هيجي جنب التاني
- بيحترم الـ padding و الـ margin [من اليمين و الشمال فقط]
- بيسمح بوجود أي عنصر قبله وبعده

Inline-Block

- بيحترم الـ padding - و الـ margin - و الـ Width - و الـ height بمعني انه بينفذ أي أمر ليهم
- بيسمح بوجود أي عنصر قبله وبعده



By: Amr Elsayed

Positioning scheme (normal flow, float, and absolute positioning).

الـ normal flow

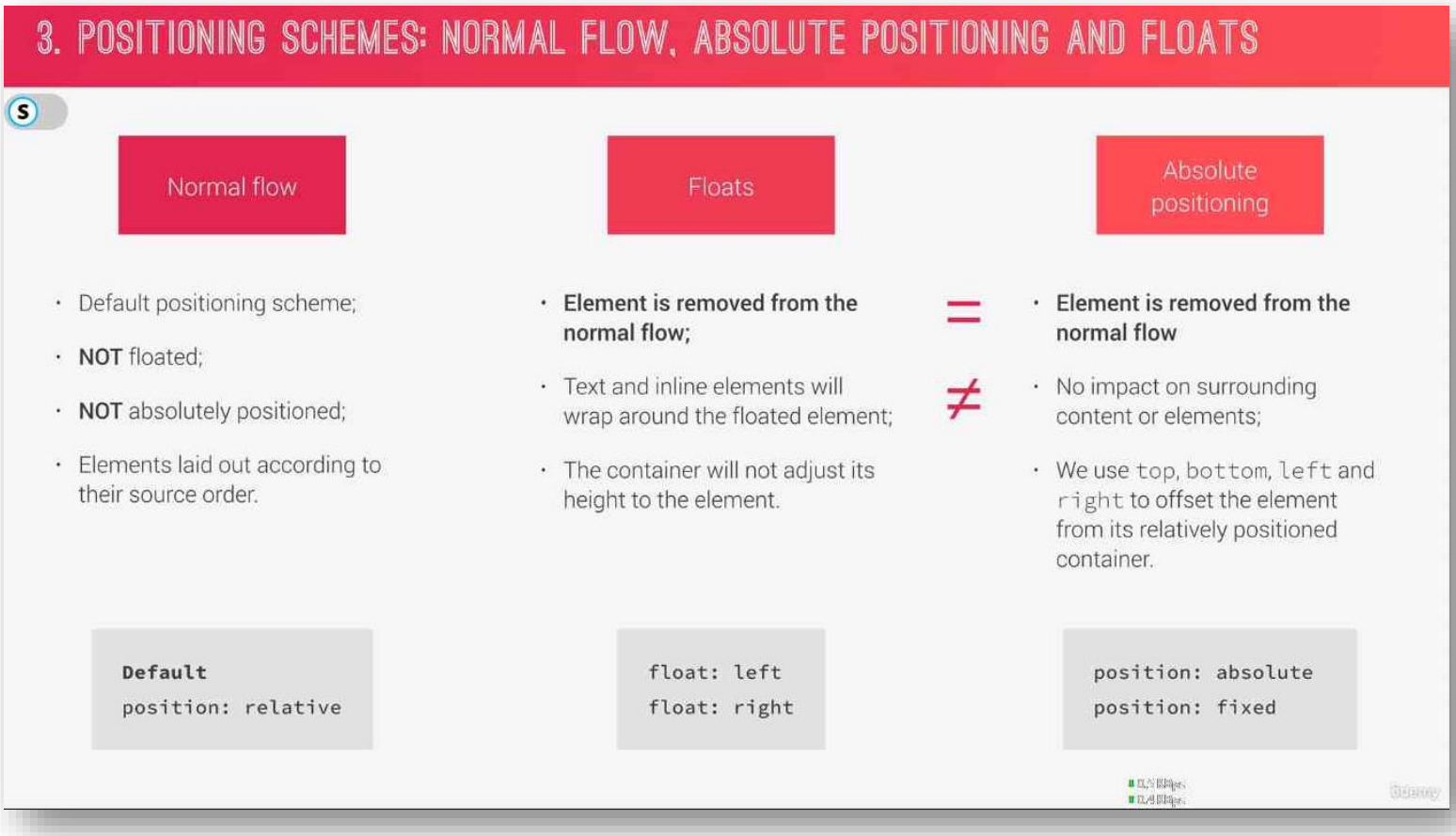
بيكون الـ default flow بتاع الـ element اللي هو position relative او position static
مش بيكون Floated ومش بيكون absolute
الـ element نفسه بيظهر بشكل الطبيعي من المصدر

الـ Floats

الـ element مع الـ Float بتشال من حالة الـ normal flow بتاعته
الـ text و الـ inline elements بتكون متحاوطة حولين الـ floated element
الـ floated element مش بيكون عندي وعي بحساب الـ height المناسب ليه او انه يفصل بينه وبين العناصر اللي تحته وفوقه لذلك بنستخدم الـ clear property بالقيمة both

الـ absolute positioning

الـ element مع الـ absolute positioning بتشال من حالة الـ normal flow بتاعته
مش بيأثر علي المحتوي ولا علي العناصر المحيطة بيه
بتقدر تستخدم معاه الـ top و الـ bottom و الـ left و الـ right عشان تحدد موقع لل element والحاجات دي بتحدد نسبة لكون الاب بتاعه relative



في الـ absolute positioning في عناصر بتيجي فوق عناصر فال CSS بتستخدم حاجة اسمها Stacking-context

By: Amr Elsayed

Stacking-context

الـ Stacking-context ببساطة شديدة هو اللي بيحدد ازاي الـ Elements بتظهر فوق بعض ازاي

ودا تعريفه بشكل دقيق

determines how elements are layered and displayed on top of each other within the document. It manages the order, visibility, and appearance of overlapping elements. Stacking contexts are created when specific CSS properties are applied to an element or its descendants.

استخدامك بقا لل Stacking-context بيخليك تعمل control علي شكل ظهور العناصر دي فوق بعض زي استخدامك لل z-index مثلا

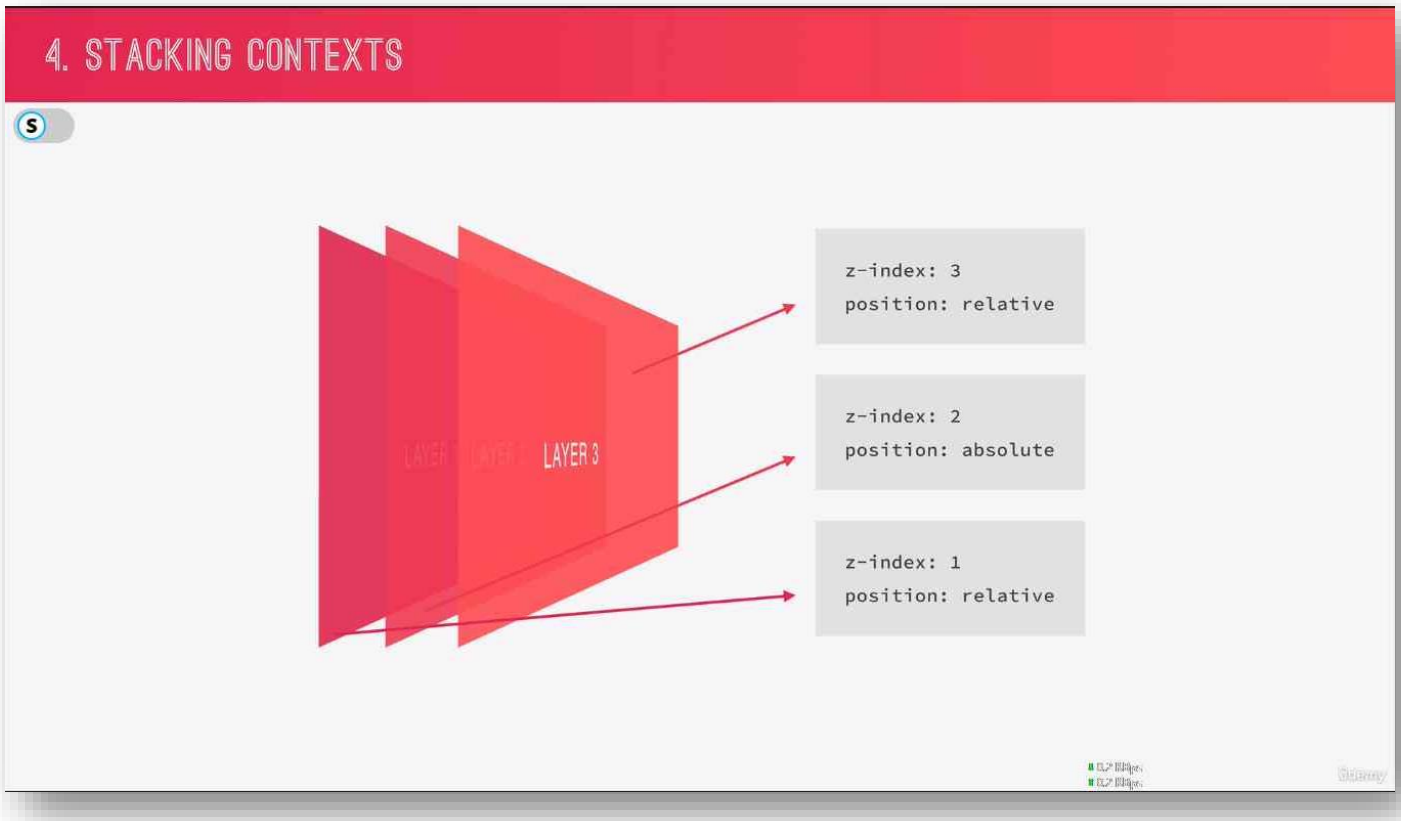
ودا تلخيص سريع ليها في تلت نقاط عن بتساعدنا ازاي وبتحل اي

Stacking contexts help solve several problems in CSS layout and rendering, including:

1- **Z-index control: Stacking contexts allow you to control the vertical stacking order of elements using the z-index property. Elements within a stacking context can be layered above or below other elements based on their assigned z-index values.**

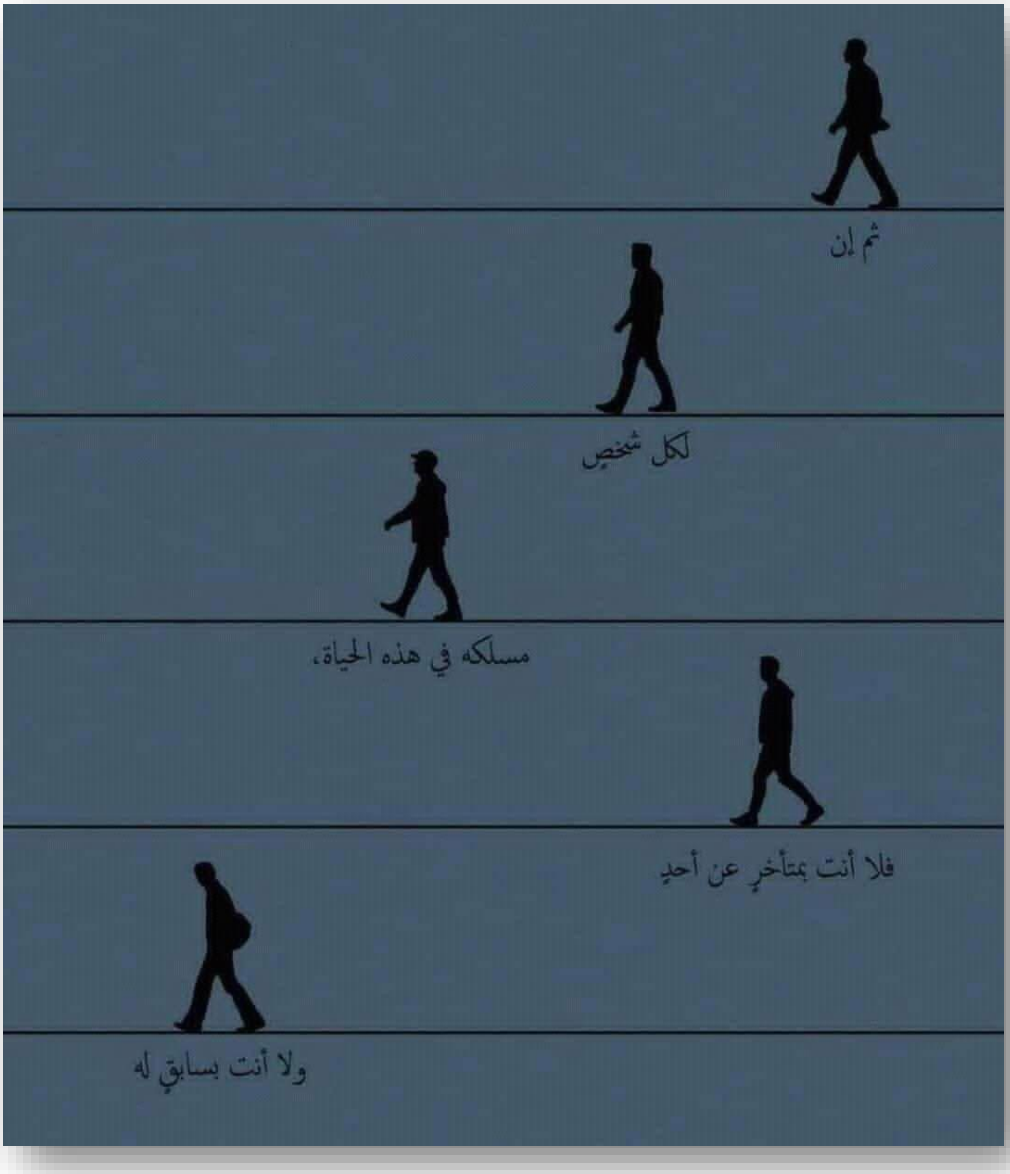
2- **Controlling overlap and visibility: Stacking contexts allow you to control the visibility and appearance of overlapping elements. Elements within different stacking contexts can be positioned and displayed independently, preventing unintended overlap or interference.**

3- **Isolation and encapsulation: Stacking contexts provide a level of isolation and encapsulation for elements and their descendants. Styles applied within a stacking context do not affect elements outside of that context, preventing unintended style cascades and conflicts.**



By: Amr Elsayed

❤ عبارات تحفيزية ❤



اذكر الله ❤

الزموا ذكر الله ففيه العون على متاعب الحياة ❤

أَلا بِذِكْرِ اللَّهِ تَطْمَئِنُّ الْقُلُوبُ

اللهم لك الحمد حمدًا طيبًا كثيرًا مباركًا فيه كما ينبغي لجلال وجهك ولعظيم سلطانك يا رب العالمين

اللهم اني أسألك علمًا نافعًا ورزقًا طيبًا وعملًا متقبلًا

لا تنسونا من دعواتكم

By: Amr Elsayed

الخاتمة وشوية فضفضة

بسم الله الرحمن الرحيم والصلاة والسلام علي أشرف المرسلين سيدنا محمد

- استكمالا للمشوار وبعد تلخيص الـ HTML و الـ CSS و الـ SASS و الـ JS وفقنا الله لعمل هذا الملخص تحت عنوان How CSS Works Behind The Scenes .. ارجوا من الله عز وجل ان يكون ذا نفع للجميع .. و ارجوا من الله عز وجل ان يجعل هذا العمل خالصًا لوجهه الكريم وان يجعله في ميزان حسناتنا جميعا – كل الملخصات هنا .. أضغط

- الملخص دا مينفعش تشوفه لو مش دارس CSS كويس وفاهم خصائص كتير فيها وبتعرف تستخدمها بشكل كويس لازم تكون فاهم CSS وطبقت بيها مشاريع كتير عشان تبقي فاهم حاجات كتير بتتكتب في النُص

- الملخصات كلها طبعا مجانية بالكامل بس لو حابب تدعمني ولو مبلغ بسيط او زي ما بيقولو تعزمني ع كوباية قهوة فدا رقمي فودافون كاش 01005074554 تقدر لو حابب وعندك قدرة تدعمني بالمبلغ اللي تحبه

- اي دعم مادي هيجيلي باذن الله هخصص منه جزء يخرج كصدقات عني وعن اللي دعموني والجزء الباقي هيكون في تطوير الحاجات اللي بشتغل عليها وشراء كورسات اكتر للتعلم وتلخيصها للجميع باذن الله

- باذن الله الملخصات الجاية هتكون عن الـ Bootstrap بالتفصيل و الـ Tailwind وبرضو هنلخص الـ Command line والـ GitHub and Git و طبعا طبعا يعني الـ TypeScript و الـ React js بإذن الله

- بعتذر عن وجود أي أخطاء املائية او أخطاء في أي كود فالكمال لله وحده ولو لقيت أي خطأ في الشرح تواصل معايا

- خصصت في الملخص دا تلت أجزاء .. الجزء الأول لذكر الله عشان متخليش حاجه ابدا تشغلك عن ذكر الله اللي قادر ينجيك ويفتحلك ابواب الرزق ويرزقك ويسهل عليك الفهم و جزئين العبارات التحفيزية و جمل وعبارات راقت لي اه هما يعتبروا سبب رئيسي في تكبير حجم الملخص، ولكن والله حبيت انهم يكسروا ملل المذاكرة والقراءة وربما يكون فيهم نفع او الهام لحد

- الملخص دا كان مصدره كورس المحاضر Jonas Schmedtmann – من Udemy واي صور فيه كانت من الـ Course .. كورس بعنوان Advanced CSS and Sass: Flexbox, Grid, Animations and More!

- مش عايزك تتضايق لاي سبب لو مفهمتش أي جزء انا حولت ابسط المفاهيم قدر المستطاع وان شاء الله هتفهم يعني هتفهم فبقترح عليك انك تعيد قراءة الحاجة دي وتجرب تكتب code بإيدك وتجرب بنفسك

- لأي حد هيوصله الكلام دا مش طالب منك غير طلب واحد وامانة عليك لازم تنفذه و هو إنك تفتكرني بدعوة طيبة لوجه الله أنا وأهل بيتي

LinkedIn Account | GitHub Account | Codepen Account

Twitter Account | Mail Me | Facebook Account

افتكروني بدعوة طيبة لوجه الله واشوفكم ان شاء الله مع حاجه أفضل وأفضل مستقبلا

شكرا لكم ❤

By: Amr Elsayed