HTML 5+1 قدرتمندتر از HTML5

10 قابلیتی که انتظار داریم در HTML6 ببینیم

انتشار نسخه رسمی HTML5 توجه طراحان و توسعه‌دهندگان را به خود معطوف ساخت. بوم گرافیکی canvas، قابلیت drag and drop، برنامه‌های وب‌آفلاین، استفاده مستقیم SVG و MathML و... همگی دلالت بر قدرت HTML5 دارند. اگر نگاهی به روند صعودی شکل‌گیری نسخه‌های مختلف HTML داشته باشیم, به این نکته خواهیم رسید که HTML5 تحولی بزرگ و اساسی را رقم زد. اما به‌رغم ویژگی‌های قدرتمند آن هنوز هم جای کار بیشتر دارد. پشتیبانی بهتر و کامل‌تر از فایل‌های چندرسانه‌ای، مدیریت بهینه بر روند بارگذاری صفحات و جلوگیری از مصرف بیهوده پهنای باند از جمله این نیازهاست.

واژه HTML5 چند سال پیش آمد و رفت، تا این که سرانجام در پایان سال 2014 میلادی به‌عنوان یک استاندارد نهایی درآمد. در این پنج سال اظهارنظرهای مختلفی در مورد HTML5 مطرح شد، دیون آلمار موسس سایت Ajaxian Web و مدیر بخش ابزارهای طراحی موزیلا، HTML5 را نسل دوم وب نامید که به‌طور کامل مورد توجه و رسیدگی قرار گرفته است. در این مدت سایت‌های مختلف آن‌را از زوایای گوناگون مورد بررسی قرار دادند.

البته این زمان تأخیر قابل پیش‌بینی بود. آن‌هایی که در مورد سرعت و نوآوری در وب صحبت می‌کنند باید از این موضوع اطلاع داشته باشند که ما هنوز به چند پروتکل قدیمی وابسته هستیم. پروتکل‌هایی که اینترنت را فلج می‌سازند. پروتکل‌هایی همچون BGP: Border Gateway Protocol. زمانی‌که هر قدم اشتباهی می‌تواند وب یا حداقل تعدادی از وب‌سایت‌ها را دچار مشکل کند؛ شما باید به کندی به سمت آن حرکت کنید. این حرکت اشتباه باعث می‌شود تا زمان قابل توجهی صرف تفکر مجدد درباره نسل بعدی از تنظیمات که قرار است در HTML پیاده‌سازی شود، صرف شود. بدون شک، هنوز بخش قابل‌توجهی از ایده‌های HTML5 جدید هستند که به آرامی راه خود را به درون سایت‌ها باز می‌کنند. هر شخصی که تجربه کار با HTML5 را داشته باشد، از این موضوع اطلاع دارد که HTML5 مجموعه‌ای از ویژگی‌های بهبود یافته است. اما از HTML6 چه انتظاری دارید؟ کدام برچسب‌ها و ویژگی‌های جدید، طراحی وب را ساده‌تر، سریع‌تر و با خطای کمتری همراه می‌کنند؟ چه قابلیت‌های جدیدی روند ساخت سریع‌تر و نرم‌تر وب‌سایت‌ها را ساده‌تر از گذشته می‌کند؟ در ادامه ده پیشنهادی که به فرآیند بهتر شدن HTML6 منجر خواهد شد را مشاهده می‌کنید.

پیشنهاد شماره 1: کنترل بیشتر روی ویدئوها

ما ممکن است هیچ‌گاه مبارزه‌ای که بر سر کدک‌های فشرده‌سازی ویدیوها در جریان است را حل نکنیم اما در عوض می‌توانیم با آن کنار بیاییم. الگوریتم‌های فشرده‌سازی مختلف ممکن است زمان زیادی را برای پیاده‌سازی صرف کنند اما آن‌ها اغلب در رقابت هستند. چه خوب می‌شد اگر توانای نظارت و کنترل بیشتر روی فریم‌های ویدیویی که روی صفحه قرار می‌گیرند را پیدا می‌کردیم. نسخه فعلی مستطیلی در اختیار ما قرار می‌دهد که یک مجموعه متوالی از فریم‌های ویدیویی آن را پر می‌کنند و ما توانایی نظارت و کنترل روی یک متن آهنگ با حاشیه‌نویسی و زیرنویسی و این چنین کارها را داریم. اما بعضی از کاربران باهوش اقدام به استفاده همزمان از دیگر اشیاء DOM کرده‌اند. mediagroup.js نتیجه این ترکیب است. با استفاده از این ترکیب می‌توان تصویر یک ویدئو داخلی را همزمان با یک ویدئو خارجی ساخته داشت. اما چه بهتر می‌شد که از مکانیزم‌های همسان‌سازی استفاده می‌کردیم؟ اگر به‌طور مثال، توانای به‌کارگیری ترکیب DOM با ویدیو را داشتیم.

```
<script src="mediagroup.js"></script>

<video class="controller" controls mediagroup="pip">

    <source src="assets/popcorntest.mp4"></source>

    <source src="assets/popcorntest.ogv"></source>

    <source src="assets/popcorntest.webm"></source>

</video>

<video mediagroup="pip">

    <source src="assets/popcorntest.mp4"></source>

    <source src="assets/popcorntest.ogv"></source>

    <source src="assets/popcorntest.webm"></source>

</video>
```

پیشنهاد شماره 2: تصاویر هم اندازه مرورگر

برای این که یک تصویر به خوبی روی صفحه‌نمایش نشان داده شود، به چه تعداد پیسکل نیاز است؟ موضوعی که از تلفن‌همراه به لپ‌تاپ و به کامپیوترهای شخصی متفاوت است. حتی تغییر اندازه یک پنجره ممکن است روی تفکیک‌پذیری اثرگذار باشد. اما برچسب‌های <img> استاندارد فقط یک SRC دریافت می‌کنند که به یک فایل تصویری اشاره می‌کند. این فایل تصویری ممکن است از تعداد زیادی پیسکل یا تعداد کمی پیکسل ساخته شده باشد که همین موضوع بر روند رندر کردن تصویر تأثیرگذار است. اگر تعداد زیادی پیکسل وجود داشته باشد، مرورگر برای نمایش عکس باید آن‌را جمع و جور کند. همین موضوع زمان و پهنای‌باند را بیهوده هدر می‌دهد. اگر تعداد پیسکل‌ها کم باشد، تصویر واضح نخواهد بود. یک پروتکل HTML خوب می‌تواند عرض یا ارتفاع موردنظر خود را پیشنهاد داده تا سرور توانایی ارائه تفکیک‌پذیری بهینه شده‌ای را داشته باشد.

پیشنهاد شماره 3: زبان‌های قابل جایگزین

اگر جاوااسکریپت را دوست دارید، مرورگر وب عالی است؛ اگر جواب منفی است که مشکل‌ساز می‌شود.
مرورگرهای استاندارد HTML فقطوفقط با جاوااسکریپت صحبت می‌کنند, اما بنا به دلایلی فرض می‌کنیم نیاز به زبانی با ترکیب نحوی text/javascript داریم که با هر برچسب و اسکریپتی سازگار می‌آید. به‌دلیل این‌که HTML 4 پیش‌فرض نیست. اگر به مستندات HTML 4 و به بخش اسکریپت آن رجوع کنید، اطلاعات جامعی در خصوص عنصر Script دریافت می‌کنید. HTML 4 پیشنهاد داد که شخصی ممکن است از ترکیب text/tcl یا text/vbscript استفاده کند اما آیا واقعا کسی از این ترکیب‌ها استفاده می‌کند؟ مایکروسافت اعلام کرد که از text/vbscript در نسخه 11 مرورگر اینترنت‌اکسپلورر پشتیبانی نمی‌کند و به همین دلیل اعلام کرد که APIهایی همچون VBArray، execScript، text/vbs، و text/vbscript دیگر در صفحات‌وب وجود نخواهند داشت. علاوه بر این، تردیدهای زیادی وجود دارد که مردم در سال های اخیر از tcl متعلق به Sun چه استفاده‌ای کرده‌اند.

نیازی نیست این‌کار به این شکل انجام شود. گوگل Dart را به آرامی به جلو می‌برد و یک صفحه‌وب با نسخه‌ای از Chromium با یک نسخه واقعی از Dart شامل این پیغام هشدار است: « آیا از Dartium به عنوان مرورگر اصلی خود استفاده نمی‌کنید؟ و Dartium برای کاربران توزیع نشده است!». اما در آینده، ما مجموعه‌ای متسحکم‌تر از زبان‌های جایگزین را خواهیم داشت که انعطاف‌پذیری و گزینه‌های طراحی بیشتری در اختیار توسعه‌دهندگان قرار می‌دهد. اگر یک پیاده‌سازی منبع باز وجود داشته باشد، آن‌را می‌توان با تمام مرورگرهای رایج هماهنگ کرد. مجبور کردن سایت‌ها به استفاده از یک زبان جایگزین برای محتوای داخلی که مخاطبان گسترده‌ای دارد مشکل است و

جاوااسکرپیت همچنان به‌طور گسترده روی وب مورد استفاده قرار می‌گیرد اما می‌تواند به گزینه خوبی برای توسعه‌های ویژه‌ای که از یک زبان تخصصی استفاده می‌کنند، منجر شود.

پیشنهاد شماره 4: پیش‌پردازه‌های جایگزین

راه حل دیگری که به بهبود فرآیند توسعه منجر می‌شود، فراتر از جاوااسکرپیت قرار دارد و اشاره به ابزارهای تبدیل زبان به جاوااسکرپیت دارد. تعدادی از توسعه‌دهندگان از مدت‌ها قبل از پیش‌پردازه‌هایی برای ترجمه زبان‌هایی از قبیل CoffeScript به جاوااسکرپیت استفاده کرده‌اند. البته در زیر این پوشش CoffeScript تفاوت زیادی با جاوااسکرپیت ندارد و بیشتر یک ترکیب نحوی alt است که آن‌را زبان متفاوتی کرده است. زبان‌های مختلفی وجود دارد که به جاوااسکرپیت می‌توانند کامپایل شوند. Parsec، IcedCoffeeScript، Coco، Scala، Ruby ، Lisp EmberScript، CoffeeScript نمونه‌ای از این موارد به‌شمار می‌روند.

زمانی‌که یک زبان به‌صورت «ترجمه دوگانه» به جاوااسکرپیت وارد می‌شود، در همان زمان کوچک می‌شود. تولید نسخه‌ای که کوچک‌تر و خواناتر باشد به راحتی روی اینترنت انتقال می‌یابد. وقتی که این عمل یک‌مرتبه در زمان توزیع انجام شود کارایی بیشتری نسبت به انجام این عمل هر زمان که کاربری آن‌را مرور می‌کند دارد. البته این کوچک‌کردن کدها دردسرهایی به‌همراه دارد. منبع‌باز بودن یکی از مزیت‌های عالی وب به‌شمار می‌رود. بازبودن این قابلیت را در اختیار ما قرار می‌دهد تا کدهای جاوااسکرپیت را که هنوز هم برای ما قابل فهم بوده، خوانده و خطاها را شناسایی کنیم. اما Cross-compiled و minified code خوانایی آن‌ها را کاهش می‌دهد و به تدریج وب از حالت openness بودن دور می‌کند. البته این تبدیل کردن مزیت‌های مختلفی را برای مرورگرها به‌همراه دارد. ماشین‌ها کمی با یکدیگر متفاوت هستند و فرآیند تبدیل می‌تواند از مزیت‌های شناخته شده‌ای از جمله اندازه حافظه، کتابخانه‌های کارت ویدئو و ... بهره ببرد. نسخه جاری HTML فرض می‌کند یک نسخه عمومی از جاوااسکرپیت موجود بوده و بهینه‌سازی این کد فرآیند مشکلی برای ماشین محلی محسوب می‌شود.

پیشنهاد شماره 5: کتابخانه‌های ضمانت‌کننده

در دنیای برنامه‌نویسی جاوااسکرپیت از میان همه کتابخانه‌های استانداردی که برای آن قرار داشت توسط کتابخانه jQuery متحول شد. اما هنوز هم بسیاری از سایت‌ها کپی‌های خود را بارگذاری و مورد استفاده قرار می‌دهند. مقدار انرژی به‌کار رفته برای بارگذاری JQuery به‌اندازه‌ای است که برای روشن کردن نور یک کشور کوچک، درمان سرطان یا هر دو این مورد می‌تواند مورد استفاده قرار گیرد. بعضی از سایت‌ها از نسخه‌های کش شده استانداردی از کتابخانه‌های اصلی که توسط شرکت‌های بزرگ همچون گوگل یا یاهو میزبانی می‌شود استفاده می‌کنند که می‌تواند پهنای باند را به‌میزان قابل توجهی ذخیره کند. اما نسخه استاندارد HTML باید بهتر از این عمل کند. اگر تعداد قابل توجهی از طراحان یک کتابخانه را به‌تصویب رسانند، آن‌گاه می‌تواند توسط مرورگر توزیع شود. این فرآیند می‌تواند به میزان قابل توجهی زمان نوسازی نسخه کش شده جی‌کوری 1.9 را ذخیره کند.

پیشنهاد شماره 6: دسترسی محافظت شده به اطلاعات تماس

اگر کاربری روی لینک درستی کلیک کند، مرورگرها اطلاعات مکانی او را به اشتراک قرار می‌دهند. چرا بیشتر نه؟ مردم اغلب می‌خواهند یک آدرس ایمیل یا شماره تلفن را در بخش اطلاعات تماس بانک اطلاعاتی دستگاه خود ثبت کنند. برای این منظور آن‌ها باید از روش کپی و برش استفاده کنند. چرا به جاوااسکرپیت اجازه ندهیم که همه این عملیات کپی و برش را انجام دهد؟ این قابلیت به‌ویژه برای دستگاه‌های همراه ایده‌آل خواهد بود. این رابط کاربری می‌تواند به مردم این امکان را دهد تا اجازه دسترسی خودکار به کدهایی که از بعضی از دامنه‌ها و نه همه می‌آید را بدهند.

پیشنهاد شماره 7: ادغام دوربین

میان دوربین‌های وب و دوربین‌های متعددی که روی تلفن‌های همراه قرار دارد، به‌ندرت پیش می‌آید کاربری از مرورگری استفاده کند که آن مرورگر امکان برقراری ارتباط با دوربین و میکروفون را نداشته باشد.W3C روشی برای اضافه کردن تصویر یا ویدئو ضبط شده به فرم‌ها ابداع کرد که تحت عنوان HTML Media Capture در سپتامبر 2014 میلادی منتشر شد. علاوه بر این، اغلب مرورگرها از نسخه‌های ویژه خود شبیه به webkitGetUserMedia استفاده می‌کنند. تصور آن ساده است. عنصر form توانایی دسترسی به مکانی روی یک دستگاه که تصاویر در آن ذخیره شده‌اند را دارد و دستگاه نیز می‌تواند کنترل بیشتری روی دوربین و نرخ ضبط داشته باشد. این قابلیت به سایت‌ها اجازه می‌دهد تا به‌طور کامل برنامه‌های تخصصی خود را داشته باشند.

پیشنهاد شماره هشت: احرازهویت سخت‌گیرانه

این امکان وجود دارد که از روش‌های سریع و سخت‌گیرانه برای احراز هویت استفاده کنیم؟ اما ساخت سخت‌افزار ایمن و مطمئن کار مشکلی خواهد بود. یک مرورگر می‌تواند بیش از این عمل کند. به جای استفاده از کوکی‌ها می‌تواند از نشانه‌های علامت، با کلیدهای جایگذاری شده استفاده کند که درون تراشه‌های محافظت شده‌ای در یک دستگاه برای پیشگیری از دسترسی و استخراج کلید محرمانه قرار گیرند. اضافه کردن یک API به مرورگر به سایت‌ها اجازه می‌دهد به درخواست‌های با امضای دیجیتال بهتر رسیدگی کنند. البته اعتماد بیش از حد روی این قابلیت می‌تواند خطرناک باشد. اما به‌عنوان یک گام جلو رونده برای عبور از کوکی‌ها و نشست‌های تصدیق هویت می‌تواند تلقی شود.

پیشنهاد شماره نه: حاشیه‌نویسی بهتر

بخش‌های نظردهی که پایه و اساس مقالات هستند فقط سرآغازی است که چگونه می‌توانیم حاشیه‌نویسی را در مقالات انجام دهیم. این حاشیه‌نویسی‌های اضافه شده با به‌کارگیری یک ساختار استاندارد می‌تواند به پاراگراف‌ها، جملات و حتی کلمات پیوند بخورد. یک نسخه پیچیده‌تر حتی امکان حاشیه‌نویسی روی تصاویر یا لحظاتی درون ویدوئوها را نیز می‌دهد. تعدادی از سایت‌ها شروع به ارائه این مدل کرده‌اند. اما استانداردسازی آن در قالب API مزیت‌های زیادی دارد که همه سایت‌ها و مرورگرها از حاشیه‌نویسی اصلی به یک شکل واحد استفاده کنند. W3C در 4 دسامبر 2014 میلادی نتیجه مطالعات و تحقیقاتی را که در این زمینه انجام داده است، منتشر کرد.

پیشنهاد شماره 10: Stronger microformats

برچسب‌های HTML بین سرتیترها، پاراگراف‌ها و شاید پاورقی‌ها تفاوت قائل شوند اما فراتر از آن کاری نمی‌کنند. چرا یک روش استاندارد برای تعیین جزئیات مرسوم ایجاد نکنیم؟ بخش‌هایی از یک آدرس یا شماره تلفن؟ مطمئنا به‌کارگیری یک برچسب استاندارد برای تعیین آدرس‌های ایمیل، زندگی را برای اسپمرها راحت‌تر می‌کند. اما یک مجموعه از برچسب‌های استاندارد سرعت خزنده‌های وب و موتورهای جستجو را افزایش می‌دهد که مزیت‌های زیادی برای ما دارد. W3C درتاریخ 29 اکتبر سال 2009 مجموعه‌ای از این microdataها که شامل تعریف آدرس‌ها، آدرس‌های معتبر، آدرس‌های مطلق، آدرس‌های نسبی، رابط HTMLCollection و... بود را به عنوان بخشی از HTML5 معرفی کرد. ما می‌توانیم استفاده جامع‌تری از این نشانه‌گذاری‌ها برای تعیین مکان‌ها، زمان‌ها، تاریخ‌ها، عناصر برای فروش، فهرست‌بندی و تمام اشکال استاندارد داده‌ها داشته باشیم.

منبع:

اینفوورلد

تاریخ انتشار:
09 بهمن 1393

نشانی منبع: https://www.shabakeh-mag.com/workshop/238