УДК 004.55

Козаченко Ю.О., Стьопкін А.В., Новіков О.О., Польська А.А.

[1] *студентка 4 курсу фізико-математичного факультету, ДВНЗ «ДДПУ»*

[2] *канд. фізико-математичних наук, доцент каф. МНМ та МНІ, ДВНЗ «ДДПУ»*

[3] *канд. фізико-математичних наук, доцент каф. математики та інформатики, ДВНЗ «ДДПУ»*

[4] *студентка 4 курсу факультету початкової, технологічної та професійної освіти, ДВНЗ «ДДПУ»*

e-mail: stepkin.andrey@rambler.ru

Створення анімації засобами CSS.

У статті розглядається проблема створення анімованих ефектів на розроблюваних web-сторінках. Проведено дослідження у напрямку алгоритмізації процесу створення анімації засобами каскадних таблиць стилів (CSS). Наведено основні інструменти, якими володіють CSS для створення анімацій різного рівня складності.

Ключові слова: *CSS, анімація, ключовий кадр.*

Вступ

Поняття Інтернету вже давно перестало бути чимось екзотичним. Щодня людство за допомогою різноманітних стаціонарних чи мобільних пристроїв, використовуючи браузери, переглядає мільйони web-сторінок. При цьому переважна більшість користувачів навіть не замислюється про те, яким чином браузер відображає цю інформацію. Однією з необхідних умов правильного відображення є написання web-сторінок спеціальною мовою розмітки гіпертексту HyperText Markup Language (HTML) [1]. HTML - це стандартна мова гіпертекстової розмітки web-документів, яка використовується у Всесвітній павутині. Розроблено HTML в кінці 80-х років минулого століття британським вченим Тімом Бернерс-Лі. За задумом це максимально проста і легка в освоєнні мова, якою змогли б користуватися люди, які не є фахівцями в області верстки та програмування. Простота мови досягалася за рахунок застосування невеликої кількості спеціальних елементів (тегів), які дозволяли без особливих зусиль отримати на виході спеціальним чином оформлений документ [2].

HTML-елементи можна поділити на категорії: 1. Метадані, призначені для повідомлення браузеру службової інформації (до цієї категорії відносяться: link; meta; script; style; title та ін.). 2. Елементи, які містяться в тілі документа і в основному призначені для обробки і форматування інформації, виведеної на екран користувача (до цієї категорії відносяться: div; form; img; input; table

та ін.). 3. Інтерактивні елементи використовуються спеціально для взаємодії з користувачем, наприклад, кнопки, посилання або поля введення тексту (до цієї категорії відносяться: a; button; textarea; та ін.). 4. Кореневі елементи, призначені для розміщення в них інших секційних елементів. Одним з них є елемент «body», тобто тіло документа. Саме в ньому розташовуються секційні елементи і заголовки [3].

Сама HTML найчастіше використовується для написання змістовної частини web-сторінки, а для опису стильового оформлення цієї частини використовуються каскадні таблиці стилів.

CSS (англ. Cascading Style Sheets) – спеціальна мова, що використовується для опису зовнішнього вигляду сторінок, написаних мовами розмітки web-документу[4].

Найчастіше CSS використовують для візуального оформлення сторінок, написаних HTML та XHTML, але формат CSS може застосовуватися до інших видів XML-документів. Специфікації CSS були створені та розвиваються Консорціумом Всесвітньої мережі.

CSS використовується авторами web-сторінок, для визначення кольорів, шрифтів та інших аспектів вигляду сторінки. Одна з головних переваг — можливість відокремити зміст сторінки (зазвичай описаний мовою HTML, XML або подібною мовою розмітки) від оформлення документу (CSS).

Майже століття тому з'явилися перші анімовані мультфільми. З розвитком технологій анімації все більше проникають в інформаційний простір та вже стали одними з головних елементів мультимедіа проектів і презентацій, та дедалі більше використовуються на web-сторінках. У наш час сучасні web-інтерфейси вимагають високого рівня інтерактивності і простоти, чого можна досягти за допомогою грамотної CSS-анімації. Тому механізми створення таких анімацій є актуальною темою для дослідження, яке має практичне спрямування.

Основна частина

Говорячи про зовнішній вигляд документа, маємо на увазі фон, стилі тексту та інших елементів, а також їх взаємне розташування на сторінці. Так що, якщо HTML повідомляє браузеру, що це за елемент, то CSS вказує йому, як оформити зовнішній вигляд цього елемента. Такий поділ досить доречний і має низку переваг, що дозволяють більш ефективно здійснювати розробку сайтів та web-додатків. Існують в CSS і можливості для створення анімаційних ефектів й різних візуальних переходів елементів з одного стану в інший, які реалізуються за рахунок набору спеціальних властивостей, що відповідають за тривалість, напрямок, кількість повторень ефектів тощо. Для того,

щоб вказати для яких саме властивостей елемента необхідно застосовувати анімаційні ефекти, використовується окреме правило @keyframes. Воно встановлює ключові кадри при анімації елемента, які представляють собою конкретні стильові властивості елемента в даний момент.

Таким чином, анімація в CSS представляє собою не що інше, як перехід від одного набору стильових властивостей елемента до іншого. У найпростішому випадку використовується два ключових кадри, тобто два набори стильових властивостей елемента, між якими і відбувається анімаційний перехід.Самі анімаційні властивості записуються окремо від @keyframes на загальних підставах з іншими правилами CSS. А для того, щоб прив'язати анімаційні властивості до конкретної анімації, тобто до правила @keyframes, використовується властивість animation-name, яка приймає в якості значень або список імен анімацій, які повинні бути застосовані до елементу, або ключове слово none, яке використовується за замовчуванням і скасовує анімацію. Імена в списку анімацій повинні перераховуватись через кому.

Перерахуємо всі доступні анімаційні властивості, які використовуються в CSS. Почнемо з animation-delay, яке встановлює час затримки перед запуском анімації. В якості значення властивість приймає час, вказаний в секундах (s) або мілісекундах (ms). Якщо вказано нульове значення часу (використовується браузером за замовчуванням), то анімація запускається без затримок. Також дозволяється використовувати від'ємні значення, але з ними потрібно бути обережним, тому що це може привести до зміни виду анімації в початковій стадії процесу. Щоб встановити тривалість одного циклу анімації, необхідно використовувати властивість animation-duration, яке також приймає в якості значення час, вказаний в секундах (s) або мілісекундах (ms). За замовчуванням використовується час рівний нулю, що означає відсутність анімації взагалі. Від'ємні значення не дозволяються.

Кількість повторень анімації можна задати за допомогою властивості animation-iteration-count, яка приймає в якості значень або позитивне число, яке вказує кількість повторень, або ключове слово infinite, що означає відтворення анімації нескінченне число разів. При цьому дозволяється використовувати не тільки цілі значення, але і дробові. Наприклад, якщо значення дорівнює 2.5, то анімація зробить два повних цикли і потім завершиться на половині третього циклу. За замовчуванням анімація відтворюється тільки один раз.

Крім кількості повторень анімації можна задати і її напрямок. Робиться це за допомогою властивості animation-direction, яка приймає значення в вигляді ключових слів:

1. *normal* – після завершення циклу, анімація скидається в початковий стан і стартує заново (використовується за замовчуванням);
2. *alternate* – після завершення циклу, анімація починає крок за кроком відтворюватися в зворотному напрямку;
3. *reverse* – анімація починається відразу з кінця циклу, виконуючи всі кроки в зворотному напрямку, а потім скидається знову в кінець циклу;
4. *alternate-reverse* – анімація починається відразу з кінця циклу, виконуючи всі кроки в зворотному напрямку, а потім починає крок за кроком відтворюватися в прямому напрямку, повертаючись в кінець циклу, який в даному випадку є стартовою точкою.

Управляти плавністю анімаціїї (швидкістю переходів від одного стану до іншого) дозволяє властивість animation-timing-function, яка використовує значення математичної функції Безьє, а також значення покрокової функції:

1. *cubic-bezier (n1, n2, n3, n4)* – поведінка анімації буде залежати від результату обчислення функції Безьє, в якості аргументів якої можна задавати чотири числа від нуля до одиниці включно;
2. *ease* – відповідає результату функції cubic-bezier (0.25,1,0.25,1); анімація прискорюється до середини, а потім сповільнюється до кінця;
3. *ease-in* – відповідає результату функції cubic-bezier (0.42,0,1,1); анімація починає повільно прискорюватися з самого початку і до кінця;
4. *ease-out* – відповідає результату функції cubic-bezier (0,0,0.58,1); анімація стартує прискорено і сповільнюється до кінця;
5. *ease-in-out* – відповідає результату функції cubic-bezier (0.42,0,0.58,1); анімація повільно стартує і повільно закінчується;
6. *linear* – відповідає результату функції cubic-bezier (0,0,1,1); постійна швидкість на всьому проміжку відтворення;
7. *steps (n, start | end)* – тут n являє собою позитивне ціле число, яке задає число кроків функції, а ключові слова визначають коли ці кроки будуть зроблені - на початку або в кінці зазначеного проміжку часу;
8. *step-start* – відповідає результату функції step (1, start); стильові властивості елемента відразу ж приймають кінцеві значення, при цьому анімація як би відсутня;
9. *step-end* – відповідає результату функції step (1, end); стильові властивості елемента знаходяться в початковому стані зазначений час, а потім стрибком приймають кінцеві значення, при цьому анімація як би відсутня.

Якщо в певний момент потрібно поставити анімацію на паузу, слід скористатися властивістю animation-play-state, яка визначає два стану анімації:

- running – анімація програється; - paused - анімація поставлена на паузу. Якщо необхідно задати відразу кілька параметрів анімації, можна використовувати універсальну властивість animation, в якій значення відповідних анімаційних властивостей перераховуються через пробіл в наступній послідовності: animation-name || animation-duration || animation-timing-function || animation-delay || animation-iteration-count || animation-direction || animation-fill-mode || animation-play-state. Якщо значення якої-небудь властивості не буде вказано, то браузер застосує значення за замовчуванням. При цьому потрібно пам'ятати, що дана універсальна властивість дозволяє прив'язати до стилю тільки одну анімацію. Вказувати через кому імена кількох анімацій не можна.

Для того, щоб створити анімацію в CSS спочатку необхідно підготувати матеріал, тобто зображення які будуть використовуватись в подальшій роботі. Для використання обраний матеріал повинен мати певний вигляд: по-перше, на картинці мають бути тільки потрібні елементи, по-друге, вже готове зображення повинно мати прозорий фон. Все це можна зробити майже в будь-якому безкоштовному графічному редакторі. В даному випадку достатньо використати редактор зображень Paint.NET. Після підготовки файлів, які будуть використані, можна приступити до основної роботи. Для роботи з HTML та CSS доречно використовувати безкоштовний спеціалізований текстовий редактор для web-розробників Adobe Brackets [5].

Необхідно створити і зберегти html-файл зі стандартною розміткою, яка відповідає стандарту HTML5. Все те, про що буде йти мова в подальшому, можна помістити в один файл CSS. Однак, щоб легше було орієнтуватися у фрагментах описуваних стилів, краще розбити його на три:

1. style.css - призначений для опису зовнішнього вигляду всіх прошарків;
2. keyframes.css - призначений для опису ключових кадрів;
3. animation.css - призначений для виклику і налаштування самих анімацій.

Банер звичайно являє собою прямокутник, в якому рухаються графічні елементи. Ефект рухомого об'єкту досягається не за рахунок руху самого об'єкту, а завдяки елементам рухомого фону. Такий підхід дозволяє домогтися нескінченної анімації. Для роботи треба підготувати набір зображень. Майже всі вони мають розширення png. Це досить зручний формат, який дозволяє використовувати різні варіанти прозорості або напівпрозорості. Кожна картинка буде розміщена в окремому прошарку, але не за допомогою тега <img>, а як фон. При верстці будемо використовувати метод вкладення прошарків. Такий стиль написання дає цілий ряд переваг: 1. Вкладені прошарки дозволяють домогтися точного позиціонування елементів один до

одного. При цьому html-код виходить максимально компактний і зручний для сприйняття. 2. Вкладеність прошарків дозволяє більш повно використовувати властивості спадкування і каскадування CSS, дозволяючи уникнути непотрібного дублювання властивостей і правил.

Анімації в CSS3 описується за допомогою так званих ключових кадрів. Ключовим називають кадр, в якому задаються зміни анімації. До одного і того ж об'єкту може бути дано відразу кілька ключових кадрів, де кожен буде відрізнятися від собі подібного якимись властивостями. В CSS такий перелік описують за допомогою правила @keyframes. Його синтаксис такий: після ключового слова @keyframes через пробіл йде ім'я анімації. Надалі це ім'я буде використано для виклику даної анімації або, простіше кажучи, її застосування до того чи іншого елементу html-сторінки. Далі, всередині фігурних дужок описуємо перелік ключових кадрів. Опис кожного починається з селектора кадру. Його, як правило, задають у відсотках і з його допомогою, описують значення анімованих властивостей в даний момент часу, який відраховується в процентному відношенні від значення тривалості одного циклу анімації. Для будь-якої анімаційної послідовності знадобляться хоча б два ключових кадри: 1. Описує зовнішній вигляд зображення на початку анімації; 2. Як воно буде виглядати в кінці. Фактично, анімація це процес плавного переходу зображення з одного стану в інший в заданому проміжку часу. Саме в зв'язку з тим, що обидва ключових кадри зустрічаються всюди, замість їх процентного запису можна використовувати спеціалізовані селектори 1. From (замість 0% для початкового моменту анімації) 2. To (замість 100% для кінцевого моменту анімації) Після кожного селектора кадру йде своя пара фігурних дужок, усередині яких перераховуються властивості елемента сторінки, що підлягають змінам. Анімувати можна тільки ті властивості, значення яких прийнято виражати числом. Якщо точніше, – дробовим числом. Наприклад, властивість ширини блоку (width) або кольору (color, background-color) можна анімувати, а ось z-index, значенням якого може бути тільки ціле число – не можна.

Якщо необхідно, щоб ця дія повторювалося, то створений запис потрібно доповнити ще однією css-властивістю - animation-iteration-count, яка визначає кількість циклів анімації. За замовчуванням, його значення дорівнює одиниці. Щоб який-небудь рух повторювався нескінченно, то замість числа, використовуємо значення, що робить анімаційний цикл безперервним. Щоб спростити запис в CSS3 можна використовувати скорочену форму, користуючись властивістю animation, яка дозволяє задати ім'я анімації, її час, кількість циклів і багато інших властивостей в один рядок.

Таким чином, для опису руху необхідно створити послідовність ключових кадрів, в якій визначені координати фону в початковий і кінцевий моменти анімації, а потім викликати її і налаштувати. Необхідно звернути увагу, що в описі ключових кадрів йдеться тільки про те, що має відбуватися, але немає конкретних вказівок на об'єкт, з яким це відбувається. Так само не сказано, скільки часу відведено на весь цикл і скільки разів він повинен повторюватися. «Прив'язка» і налаштування ключової анімації до елементу сторінки здійснюється в css-правилі, що описує зовнішній вигляд конкретного селектора. Такий принцип, дуже практична річ, тому що одного разу описану послідовність ключових кадрів можна згодом неодноразово підключати до різних елементів сторінки.

Якщо анімація повинна повторюватися нескінченно, то протікати вона повинна рівномірно. За це в CSS відповідає властивість animation-timing-function, яка має цілий ряд значень, які забезпечують плавність перебігу анімації. Існують наступні функції пом'якшення:

1. *linear* анімація протікає рівномірно протягом усього часу виконання;
2. *ease* анімація відбувається спочатку повільно, потім починає прискорюватися і знову сповільнюється до кінця. (Ця функція якраз і є використовуваної за замовчуванням);
3. *ease-in* анімація протікає повільно на початку, а потім прискорюється;
4. *ease-out* анімація сповільнюється до кінця;
5. *ease-in-out* анімація протікає повільно на початку і в кінці;
6. *cubic-function (x, x, x, x)* універсальна функція, яка дозволяє задати різну швидкість виконання анімації за допомогою числових значень. Строго кажучи, перераховані вище функції, є окремими випадками cubic-function.

Це список лише тих значень, які використовуються найчастіше. Насправді, їх набагато більше. У конкретному випадку підходить перше значення, яке забезпечує рівномірність руху на протязі всього часу анімації.

В CSS3 є властивість animation-direction і з її допомогою можна встановити порядок виконання анімації. Воно може приймати одне з трьох значень:

1. *normal* - використовується за умовчанням і означає, що анімація відбувається в тому порядку, в якому вона описана в ключових кадрах;
2. *reverse* - анімація відбувається в зворотному порядку, тобто останній ключовий кадр вважається першим, а перший навпаки - останнім;
3. *alternate* - чергування анімації, тобто анімація повинна виконуватися в зворотному порядку в парні рази і в нормальному в непарні.

Висновки

Одним із способів пожвавити контент в HTML є використання так званих анімацій, створених засобами CSS. Принцип їх роботи досить простий. Вони надають можливість анімувати властивості CSS тих елементів, які знаходяться під їх впливом. Це дозволяє створювати різноманітні анімаційні ефекти, наприклад, змушувати об'єкти рухатися, зникати і з'являтися, змінювати колір тощо. Але для того, щоб створювати щось складне, подібне руху об'єктів, треба розвивати власне уявлення про анімаційні перетворення.

Як правило, стандартне використання анімації полягає в тому, щоб змінювати якісь елементів сайту плавно з плином часу. Але чим динамічніше і сучасніше анімація, тим більше уваги користувачів ви привернете до свого проекту. Крім того, важливу роль відіграє також і інтерактивність.

Література

1. *Шафер С.* HTML, XHTML и CSS. Библия пользователя / С. Шафер. — М.: Вильямс, 2010. — 656 с.
2. *Лабберс К.* HTML5 для профессионалов: мощные инструменты для разработки современных веб-приложений / К. Лабберс, Н. Олберс, К. Салим. — М.: Вильямс, 2011. — 272 с.
3. *Пасічнік О.Г.* Основи веб-дизайну / О.Г. Пасічнік, О.В. Пасічнік, І.В. Стеценко. — Вид. група BHV, 2009. — 336 с.
4. *Макфарланд Д.* Большая книга CSS3. 3-е изд. / Д. Макфарланд. — СПб.: Питер, 2014. — 608 с.
5. *Стьопкін А.В.* Використання редактору Brackets на уроках інформатики / А.В. Стьопкін, Т.В. Турка, М.С. Кузюра // Духовність особистості: методологія, теорія і практика : збірник наукових праць / гол. редактор Г.П. Шевченко. — Вип. 5 (80). — Сєвєродонецьк: вид-во СНУ ім. Даля, 2017. — С. 218–223.

Kozachenko Yu.O., Stopkin A.V., Novikov O.O., Polska A.A.
Donbas State Pedagogical University, Sloviansk, Ukraine.

Creating animations using CSS tools.

The article considers the problem of creating animated effects on the developed web pages. A study has been carried out in the direction of algorithmizing the process of creating animation using the means of cascading style sheets (CSS). The main tools that CSS has for creating animations of different levels of complexity are given.

Keywords: *CSS, animation, key frame.*