**Программный каркас для создания спрайтовой анимации на HTML5**

А.С. Лизин, О.К. Архангельский, И.В. Бабаков

Научный руководитель Ф.Е. Татарский, ассистент каф. АИКС, НИТПУ

sogimu@nxt.ru

Национальный исследовательский Томский политехнический университет

С появлением поддержки нового, но еще не готового стандарта HTML(HTML5) ведущими производителями браузерных ядер (WebKit (Chrome), Gecko(FireFox), Presto(Opera)) и отказом фирмы Adobe от развития flash player для мобильных[1] и linux[2] платформ, перед разработчиком интерактивных веб-приложений встает необходимость использования графических возможностей HTML5.

При использовании HTML5 создание графики возможно с помощью:

1. API 2d-context[3] - реализует работу с растровой графикой;
2. SVG[4] - реализует работу с векторной графикой.

Для выбора технологии был использован сравнительный тест 2d-contex и SVG [5]: тест состоял в отслеживании числа кадров в секунду (FPS) при изменении количества объектов (движущиеся с максимальной скоростью квадраты) на экране при использовании 2d-context и при использовании SVG. Результаты приведены на рисунке1.

Как можно видеть на графике (Рисунок 1), 2d-contex более производителен при большом количестве объектов и так как основными задачами, решаемыми с помощью графических возможностей HTML5 и Flash, чаще всего являются реализация спрайтовой анимация и работа с изображениями, можно сделать вывод, что для замены технологии Flash больше всего подходит 2d-context ведь он, как и Flash, реализует работу с растровой графикой и позволяет работать с большим числом объектов. Очевидно, что 2d-context должен оказаться наиболее подходящей альтернативной Flash.



Рисунок 1 — производительность отрисовки при использовании Canvas и SVG;

Однако использование 2d-context влечет за собой следующие трудности:

1. неготовый стандарт. Стандарт в стадии тестирования – W3C объявил о планах, согласно которым окончательная версия стандарта HTML5 будет утверждена лишь к 2014 году[6];
2. отсутствие визуальных сред, вроде Flash Professional CS6;
3. слабое развитие специализированных каркасов, вызванные, скорее всего, незавершенностью стандарта;
4. разработка без использования специализированного каркаса требует большой объем кода для реализации несложной графики.

Но тем не менее, не смотря на трудности, HTML5 перспективная технология, т.к. не зависит от какой то одной компании, имеет открытый стандарт и рекомендуется такими компаниями как Apple, Microsoft.

Одним из вариантов решения описанных выше проблем может быть создание каркаса, сравнимого по гибкости с разработкой на API и позволяющего уменьшать количество кода за счет готовой реализации некоторых программных решений (# оптимизация по времени прорисовки, готовый цикл анимации и т.д.). Каркас призван облегчить труд разработчика и дать возможность сосредоточить свое внимание на реализации конкретного приложения.

Для устранения выше озвученных трудностей было принято решение изготовить каркас. Был разработан каркас, вводящий:

1. обертки для графических примитивов (врапперы), дающие возможность ассоциировать примитив с отдельной сущностью, настраивая внешний вид через набор свойств и методов;

*Обертки позволили уменьшить количество кода для одного примитива.*

1. вводить абстракцию объектов, поддерживающих наследование, что дает возможность описывать графические объекты (кнопки, элементы интерфейса и т.д.) через описание класса, наследующего класс Object;

*Наследование и объекты дали возможность частично использовать концепцию ООП.*

1. объекты с древовидной структурой, дающие возможность структурировать объекты из объектов и примитивов;

*Возможность структурирования позволила уменьшать сложность работы с множеством объектов за счет возможности манипулировать объектами объектов.*

1. систему событий с возможностью привязки своих обработчиков к таким событиям, как загрузка зависимостей объекта (изображения, музыка и т.д.), щелчок курсора, нажатия клавиш клавиатуры и т.д.;

*Система событии позволила управлять объектами с помощью мыши и клавиатуры*

1. слои для размещения объектов;

*При размещении объектов сцены на небольшом количестве слоев, можно получать выигрыш в производительности.*

В заработанном каркасе реализованы далеко не все возможные виды упрощения работы. Например возможно существенно оптимизировать процесс прорисовки, через перерисовку только тех объектов чей внешний вид изменился. Ознакомится с каркасом можно по ссылке: <https://github.com/sogimu/ArmLib>.

**Список литературы:**

1. An Update on Flash Player and Android: Adobe AIR and Adobe Flash Player Team Blog. — 28.06.2012 [Электронный ресурс]. — URL: <http://blogs.adobe.com/flashplayer/2012/06/flash-player-and-android-update.html> (дата обращения: 03.03.2013).
2. Adobe and Google Partnering for Flash Player on Linux: Adobe AIR and Adobe Flash Player Team Blog. — 28.06.2012 [Электронный ресурс]. — URL: http://blogs.adobe.com/flashplayer/2012/02/adobe-and-google-partnering-for-flash-player-on-linux.html (дата обращения: 03.03.2013).
3. HTML Canvas 2D Context: World Wide Web Consortium. — 17.12.2012 [Электронный ресурс]. — URL: <http://www.w3.org/TR/2dcontext/> (дата обращения: 04.03.2013).
4. Scalable Vector Graphics (SVG) 1.1 (Second Edition): World Wide Web Consortium. — 16.08.2011 [Электронный ресурс]. — URL: <http://www.w3.org/TR/SVG/> (дата обращения: 04.03.2013).
5. SVG vs Canvas Performance: Joel Oughton. — 03.04.2011 [Электронный ресурс]. — URL: <http://joeloughton.com/blog/web-applications/svg-vs-canvas-performance/> (дата обращения: 04.03.2013).
6. W3C Confirms May 2011 for HTML5 Last Call, Targets 2014 for HTML5 Standard: World Wide Web Consortium. — [Электронный ресурс]. — URL: <http://www.w3.org/2011/02/htmlwg-pr.html> (дата обращения: 02.03.2013).