***D3.js***

D3.js это JavaScript библиотека для обработки и визуализации данных. D3 позволяет работать с данными, используя HTML, CVG и CSS.

Для подключения библиотеки нужно добавить:

*<script src="https://d3js.org/d3.v4.min.js"></script>*

**Вступление**

**D3** позволяет связать произвольные данные с DOM, а затем применить преобразования данных к документу. Например, можно использовать D3 для создания таблицы HTML из массива чисел, или использовать те же данные для создания интерактивной SVG гистограммы с плавными переходами и взаимодействием.

D3 не стремится обеспечить все возможные функции. Вместо этого, D3 дает возможность эффективно манипулировать документами, основанными на данных. Это позволяет избежать проприетарного представления и дает исключительную гибкость, открывая все возможности веб-стандартов, таких как HTML, SVG и CSS. При этом с минимальными затратами, D3 чрезвычайно быстр, поддерживает большие наборы данных и динамическое поведение для взаимодействия и анимации. Функциональный стиль D3 позволяет повторно использовать код через разнообразную коллекцию компонентов и плагинов.

**Selections**

Изменение документов с использованием [API W3C DOM](http://www.w3.org/DOM/DOMTR) является утомительным: имена методов многословны, императивный подход требует ручной итерации и ведение учета временного состояния. Например, изменим цвет текста абзаца элементов:

*var paragraphs = document.getElementsByTagName("p");*

*for (var i = 0; i < paragraphs.length; i++) {*

*var paragraph = paragraphs.item(i);*

*paragraph.style.setProperty("color", "white", null);*

*}*

D3 использует декларативный подход, работающий на произвольном множестве узлов, называемых *selections*. Например, можно переписать предыдущий цикл так:

*d3.selectAll("p").style("color", "white");*

При этом можно манипулировать отдельными узлами по мере необходимости:

*d3.select("body").style("background-color", "black")*

*;*

Селекторы определяются W3C Selectors API и изначально поддерживаются современными браузерами. Обратная совместимость для старых браузеров может быть предоставлена Sizzle (JavaScript Selector Library). Элементы могут быть выбраны с помощью различных предикатов, включая защитную оболочку, значения атрибутов, класса и ID.

D3 предоставляет множество методов изменения узлов: установка атрибутов или стилей; добавление, удаление или сортировка узлов; изменение HTML или текстового содержимого. Это достаточно для подавляющего большинства потребностей. Также есть прямой доступ к лежащей в основе DOM, поскольку каждый selection D3 это просто массив узлов.

**Динамические свойства:**

D3 имеет сходство с другими структурами DOM, такими как jQuery или Prototype. Тем не менее, стили, атрибуты и другие свойства могут быть определены как функции данных в D3 (не только простые константы). Несмотря на кажущуюся простоту, эти функции могут быть удивительно мощным. Например, функция d3.geo.path.

D3 предоставляет множество встроенных повторно используемых функций и функциональных заводов, таких как графические примитивы для области, линии и круговые диаграммы.

Например, зададим случайным образом цвет параграфов:

*d3.selectAll("p").style("color", function() {*

*return "hsl(" + Math.random() \* 360 + ",100%,50%)";*

*});*

Можно задать альтернативные оттенки серого для четных и нечетных узлов:

*d3.selectAll("p").style("color", function(d, i) {*

*return i % 2 ? "#fff" : "#eee";*

*});*

Рассчитанные свойства часто ссылаются на связанные данные. Данные задаются в виде массива значений, и каждое значение передается в качестве первого аргумента ( d) к функциям выбора. С помощью join-by-index по умолчанию первый элемент в массиве данных передается в первый узел в selection, второй элемент ко второму узлу, и так далее.

Например, если связать массив чисел с параграфом элементов, то можно использовать эти номера для вычисления динамических размеров шрифта:

d3.selectAll("p")

.data([4, 8, 15, 16, 23, 42])

.style("font-size", function(d) { return d + "px"; });

После того как данные были связаны с документом, можно опустить data operator; D3 будет извлекать ранее связанные данные. Это позволяет пересчитывать свойства без подмены.

**Enter and Exit**

Используя D3 enter and exit selection, можно создавать новые узлы для входящих данных и удалять исходящие узлы, которые больше не нужны.

Когда данные привязаны к selection, каждый элемент в массиве данных в паре с соответствующим узлом в выделенном selection. Если узлов меньше чем данных, то дополнительные элементы данных формируются с помощью enter selection.

Например:

*d3.select("body")*

*.selectAll("p")*

*.data([4, 8, 15, 16, 23, 42])*

*.enter().append("p")*

*.text(function(d) { return "I’m number " + d + "!"; });*

Обновление узлов selection по умолчанию результат data оператора. Таким образом, если забыть о enter and exit selection, то автоматически будут выбраны только те элементы, для которых существует соответствующие данные. Довольно часто разбивают первоначальный selection на три части: обновление узлов для изменения, entering nodes для добавления, и exiting nodes для удаления.

*// Update…*

*var p = d3.select("body")*

*.selectAll("p")*

*.data([4, 8, 15, 16, 23, 42])*

*.text(function(d) { return d; });*

*// Enter…*

*p.enter().append("p")*

*.text(function(d) { return d; });*

*// Exit…*

*p.exit().remove();*

Реализуя эти три случая по отдельности, необходимо указать, какие именно операции выполняются на каких узлах. Это повышает производительность и дает больший контроль над переходами. Например, с помощью гистограммы можно инициализировать въезжающие полоски с использованием старой шкалы.

D3 позволяет преобразовывать документы на основе данных, включая создание и удаление элементов. D3 позволяет изменить существующий документ в ответ на взаимодействие пользователя. Дает возможность сделать анимацию, или даже асинхронное уведомление от третьей стороны. Гибридный подход возможен даже, если документ инициализирован на сервере, и обновляется на клиенте через D3.

**Transformation, Not Representation**

D3 не вводит новое визуальное представление. В отличие от Processing, Raphaël, или Protovis, графических знаки D3 поступают непосредственно из веб-стандартов: HTML, SVG и CSS. Например, можно создать SVG элементы, используя D3 и их стиль с внешними таблицами стилей. Можно использовать композитные эффекты фильтра. Если производители браузеров введут новые функции, то будет возможность использовать их (не требуется обновления).

Лучше всего, дебажить D3, используя the browser’s built-in element inspector.

**Transitions**

D3 сфокусирована на естественной трансформации анимационных переходов. Переходы постепенно интерполируют стили и атрибуты с течением времени. Интерполяторы D3 поддерживают числа и числа, встроенные внутри строк (размеры шрифта, данные пути, и т.д.), а также составные значения. Можно даже расширить интерполятора реестра D3 для поддержки сложных свойств и структуры данных.

Например, поменять размытый фон страницы на черный:

*d3.select("body").transition()*

*.style("background-color", "black");*

Изменить размер кругов в символе карты с пораженной задержкой:

*d3.selectAll("circle").transition()*

*.duration(750)*

*.delay(function(d, i) { return i \* 10; })*

*.attr("r", function(d) { return Math.sqrt(d \* scale); });*

Модифицируя только те атрибуты, которые фактически изменяются, D3 снижает накладные расходы и обеспечивает большую графическую сложность при высокой частоте кадров. D3 также позволяет использовать CSS3 переходы; D3 не заменяет набор инструментов браузера, но выставляет его в пути, который проще в использовании.