деревья

хорошо

битовая карта

Игорь Алексеенко

училка в HTML Academy

То, что мы разрабатываем, скорее, уже не сайты, а RIA

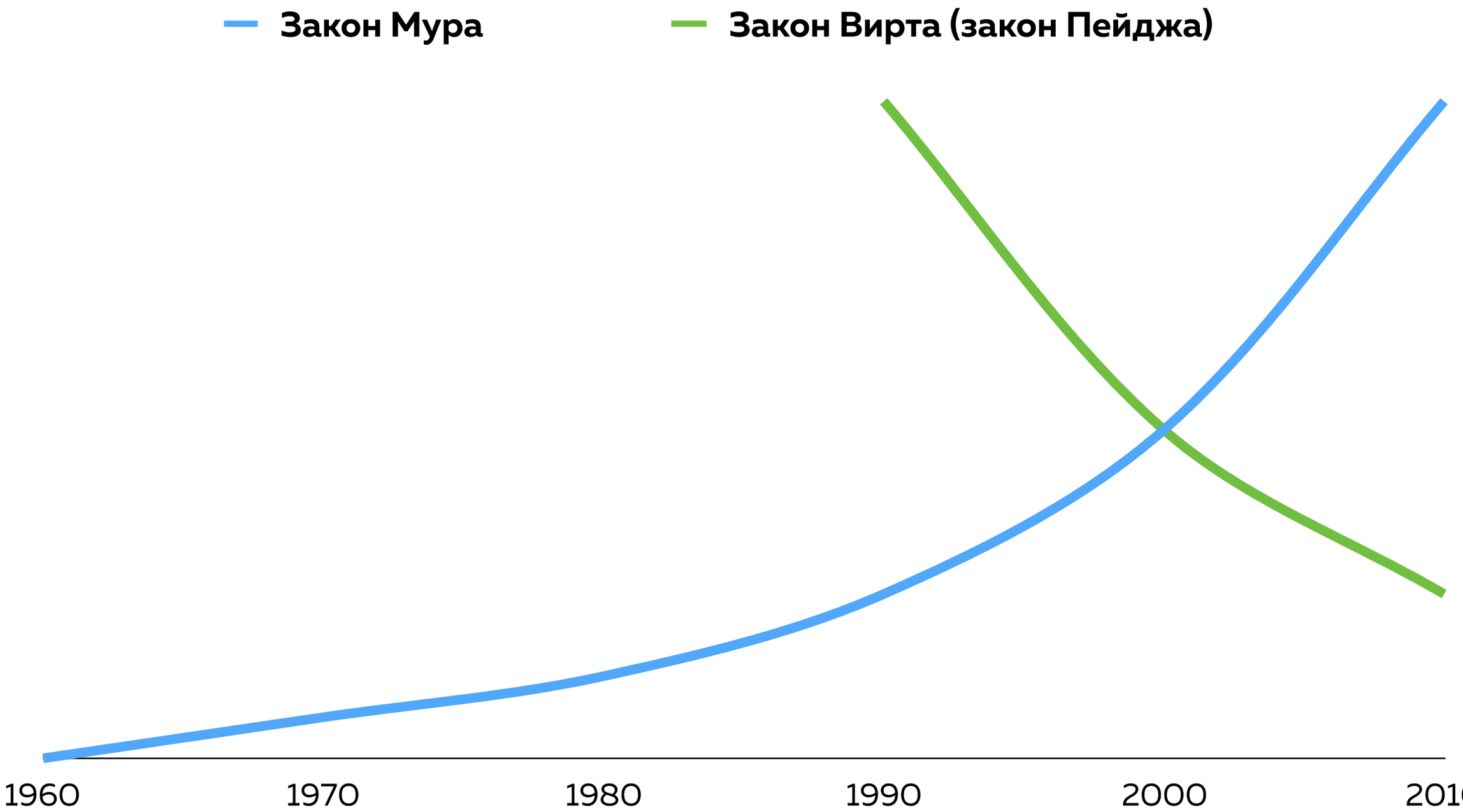
RIA —

(Rich Internet Application, сербохорв. насыщенное интернет-приложение), иногда IIA — Installable Internet Application — сайт, обладающий свойствами настольного приложения

Многие современные проблемы фронтендеров были решены еще в 80-х 😎📼💾☎️👾🚀

просто мы об этом не знаем 👶 👶 👶

Законы развития ПО

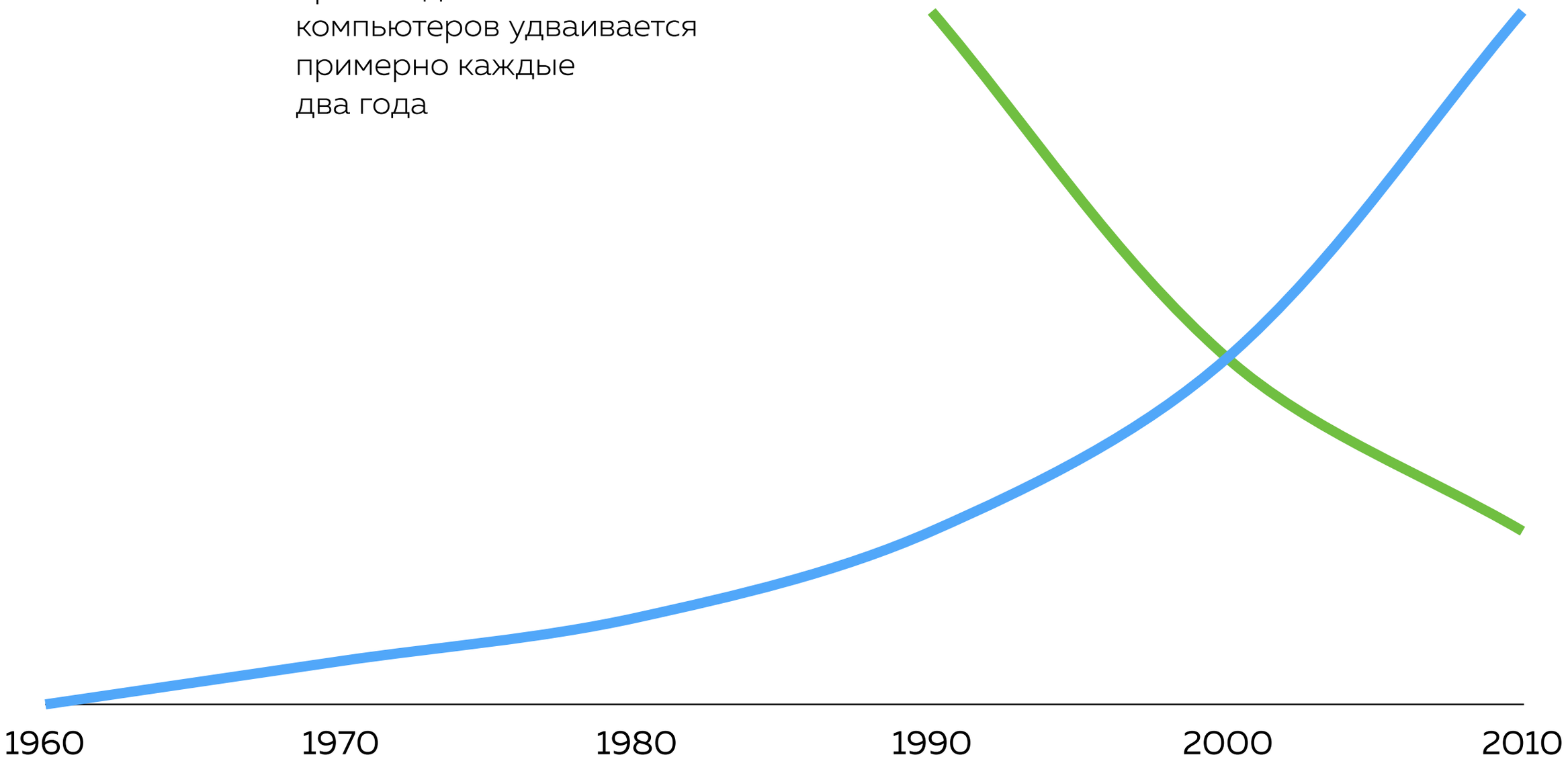


Законы развития ПО

Закон Мура

производительность компьютеров удваивается примерно каждые два года

Закон Вирта (закон Пейджа)



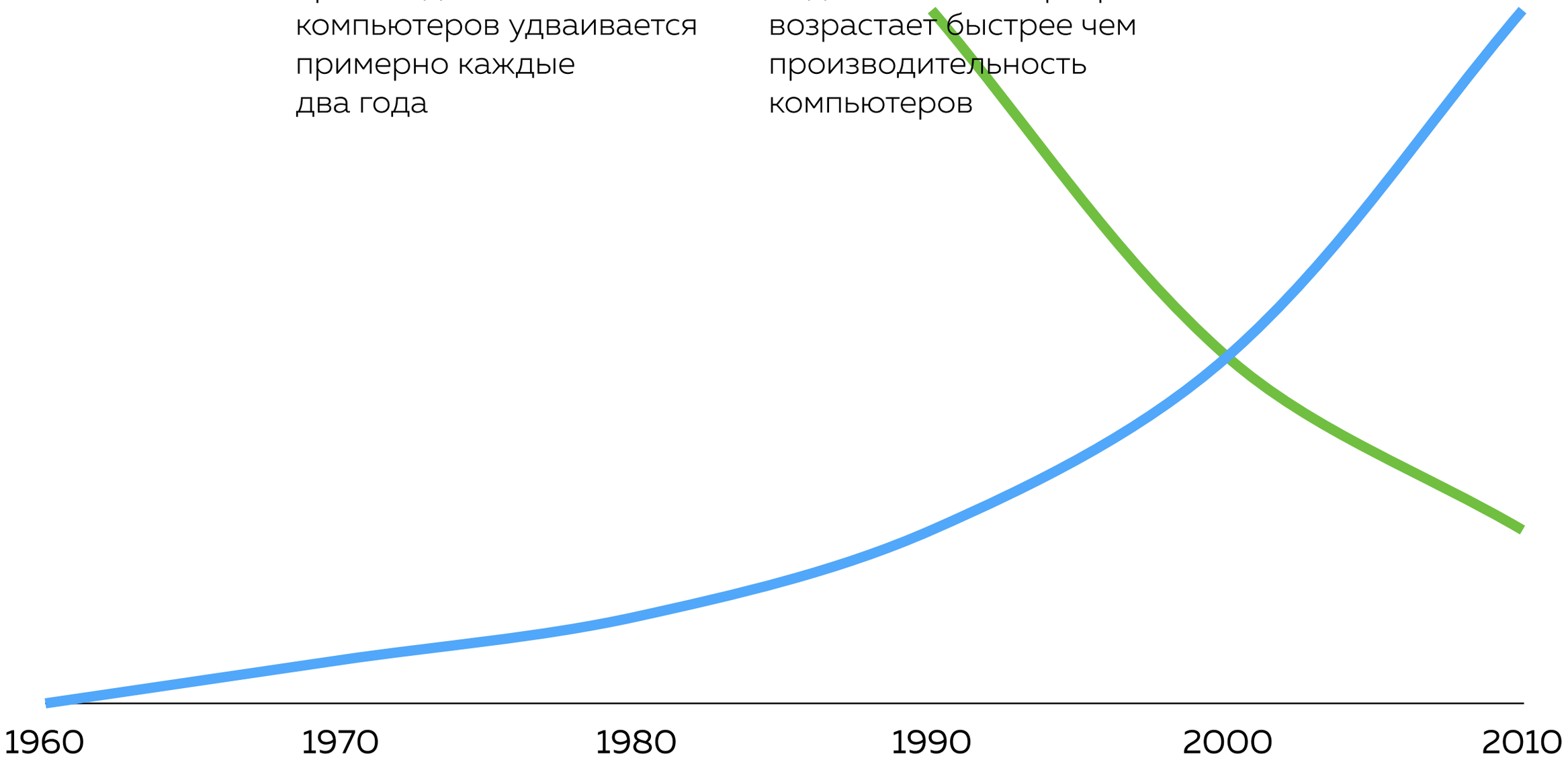
Законы развития ПО

Закон Мура

производительность компьютеров удваивается примерно каждые два года

Закон Вирта (закон Пейджа)

медлительность программ возрастает быстрее чем производительность компьютеров



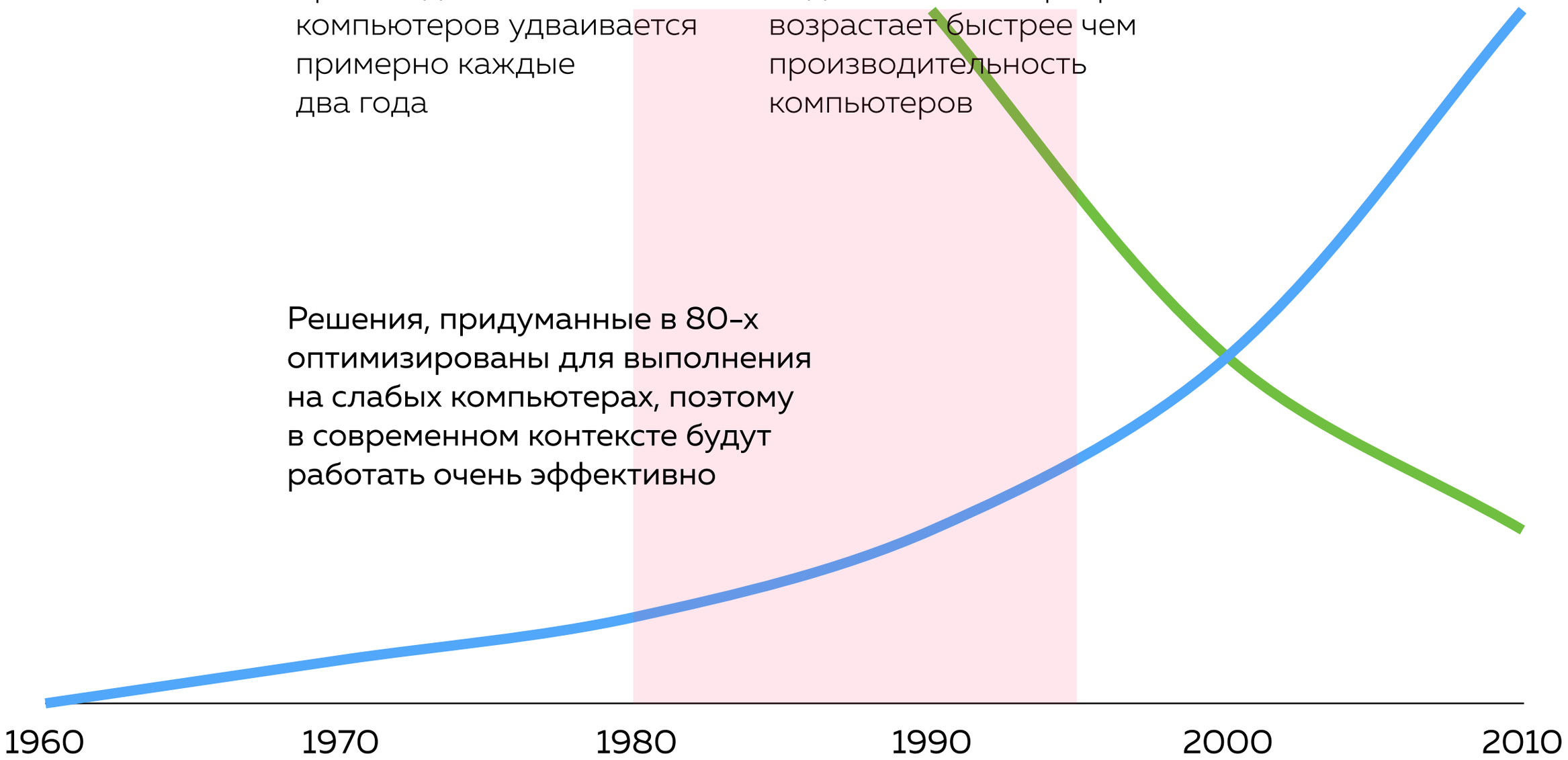
Законы развития ПО

Закон Мура

производительность компьютеров удваивается примерно каждые два года

Закон Вирта (закон Пейджа)

медлительность программ возрастает быстрее чем производительность компьютеров



Практические примеры

повседневные задачи, решенные за три копейки, заодно новенькими (хорошо забытыми старыми) способами

Практические примеры

повседневные задачи, решенные за три копейки, заодно новенькими (хорошо забытыми старыми) способами

- хранение множественных состояний

Практические примеры

повседневные задачи, решенные за три копейки, заодно новенькими (хорошо забытыми старыми) способами

- хранение множественных состояний
    - UX-компоненты

Практические примеры

повседневные задачи, решенные за три копейки, заодно новенькими (хорошо забытыми старыми) способами

- хранение множественных состояний
  - UX-компоненты
  - объекта в игре

Практические примеры

повседневные задачи, решенные за три копейки, заодно новенькими (хорошо забытыми старыми) способами

- хранение множественных состояний
  - UX-компоненты
  - объекта в игре
- передача нескольких флагов одновременно

Практические примеры

повседневные задачи, решенные за три копейки, заодно новенькими (хорошо забытыми старыми) способами

- хранение множественных состояний
  - UX-компоненты
  - объекта в игре
  - передача нескольких флагов одновременно
- хранение и передача состояния сложного лейаута

Практические примеры

повседневные задачи, решенные за три копейки, заодно новенькими (хорошо забытыми старыми) способами

- хранение множественных состояний
  - UX-компоненты
  - объекта в игре
  - передача нескольких флагов одновременно
- хранение и передача состояния сложного лейаута
- зависимая форма

Практические примеры

повседневные задачи, решенные за три копейки, заодно новенькими (хорошо забытыми старыми) способами

- хранение множественных состояний
  - UX-компоненты
  - объекта в игре
    - передача нескольких флагов одновременно
- хранение и передача состояния сложного лейаута
- зависимая форма
- одна популярная библиотека

Практические примеры

повседневные задачи, решенные за три копейки, заодно новенькими (хорошо забытыми старыми) способами

- хранение множественных состояний
  - UX-компоненты
  - объекта в игре
  - передача нескольких флагов одновременно
- хранение и передача состояния сложного лейаута
- зависимая форма
- одна популярная библиотека

Практические примеры

повседневные задачи, решенные за три копейки, заодно новенькими (хорошо забытыми старыми) способами

- хранение множественных состояний
  - UX-компоненты
  - объекта в игре
  - передача нескольких флагов одновременно
- хранение и передача состояния сложного лейаута
- зависимая форма
- одна популярная библиотека

одна технология

другая

Кнопка


Кнопка


Кнопка

Кнопка

Кнопка


Кнопка

component.js — /Users/igoralexeenko/dev/ha/demo.js

component.js

```
1   'use strict';
2
3   class UXComponent {
4     constructor() {
5       this.disabled = false;
6     }
7   }
8
```

Line 0 | File 0 | Project 0 | No Issues | component.js | 5:27 | LF | UTF-8 | JavaScript | master | +8

component.js — /Users/igoralexeenko/dev/ha/demo.js

component.js

```
1   'use strict';
2
3   class UXComponent {
4     constructor() {
5       this.disabled = false;
6       this.focused = false;
7     }
8   }
9
```

Line 0 | File 0 | Project 0 | No Issues | component.js | 6:26 | LF | UTF-8 | JavaScript | master | +9

component.js — /Users/igoralexeenko/dev/ha/demo.js

component.js

```
1   'use strict';
2
3   class UXComponent {
4     constructor() {
5       this.disabled = false;
6       this.focused = false;
7       this.hovered = false;
8     }
9   }
10
```

Line 0 | File 0 | Project 0 | No Issues | component.js | 7:26 | LF | UTF-8 | JavaScript | master | +10

component.js — /Users/igoralexeenko/dev/ha/demo.js

component.js

```
1   'use strict';
2
3   class UXComponent {
4     constructor() {
5       this.disabled = false;
6       this.focused = false;
7       this.hovered = false;
8       this.active = false;
9     }
10  }
11
```

Line 0 | File 0 | Project 0 | No Issues | component.js | 8:25 | LF | UTF-8 | JavaScript | master | +11

component.js — /Users/igoralexeenko/dev/ha/demo.js

component.js

```
1   'use strict';
2
3   class UXComponent {
4     constructor() {
5       this.disabled = false;
6       this.focused = false;
7       this.hovered = false;
8       this.active = false;
9       this.hasIcon = false;
10    }
11  }
12
```

Line 0 | File 0 | Project 0 | No Issues | component.js | 9:26 | LF | UTF-8 | JavaScript | master | +12

component.js — /Users/igoralexeenko/dev/ha/demo.js

component.js

```
1   'use strict';
2
3   class UXComponent {
4     constructor() {
5       this.disabled = false;
6       this.focused = false;
7       this.hovered = false;
8       this.active = false;
9       this.hasIcon = false;
10    }
11
12    getClassName() {
13
14    }
15  }
16
```

Line 0 File 0 Project 0 ✔ No Issues component.js 13:1 LF UTF-8 JavaScript master +16

component.js — /Users/igoralexeenko/dev/ha/demo.js

component.js

```
1   'use strict';
2
3   class UXComponent {
4     constructor() {
5       this.disabled = false;
6       this.focused = false;
7       this.hovered = false;
8       this.active = false;
9       this.hasIcon = false;
10    }
11
12    getClassName() {
13      let classname = [];
14    }
15  }
16
```

Line 0 File 0 Project 0 ✔ No Issues component.js 13:24 LF UTF-8 JavaScript master +16

component.js

```
1   'use strict';
2
3   class UXComponent {
4     constructor() {
5       this.disabled = false;
6       this.focused = false;
7       this.hovered = false;
8       this.active = false;
9       this.hasIcon = false;
10    }
11
12    getClassName() {
13      let classname = [];
14    }
15  }
16
```

component.js — /Users/igoralexeenko/dev/ha/demo.js

component.js

```
'use strict';

class UXComponent {
  constructor() {
    this.disabled = false;
    this.focused = false;
    this.hovered = false;
    this.active = false;
    this.hasIcon = false;
  }

  getClassName() {
    let classname = [];
    if (this.disabled) {
      classname.push('item-disabled');
    }
  }
}
```

Line 0 File 0 Project 0 ✔ No Issues component.js 15:39 LF UTF-8 JavaScript master +19

component.js

```
'use strict';

class UXComponent {
  constructor() {
    this.disabled = false;
    this.focused = false;
    this.hovered = false;
    this.active = false;
    this.hasIcon = false;
  }

  getClassName() {
    let classname = [];
    if (this.disabled) {
      classname.push('item-disabled');
    }
    if (this.focused) {
      classname.push('item-focused');
    }
  }
}
```

component.js

```
'use strict';

class UXComponent {
  constructor() {
    this.disabled = false;
    this.focused = false;
    this.hovered = false;
    this.active = false;
    this.hasIcon = false;
  }

  getClassName() {
    let classname = [];
    if (this.disabled) {
      classname.push('item-disabled');
    }
    if (this.focused) {
      classname.push('item-focused');
    }
    if (this.hovered) {
      classname.push('item-hovered');
    }
    if (this.active) {
      classname.push('item-active');
    }
    if (this.hasIcon) {
      classname.push('item-hasicon');
    }

  }
}

```

component.js

```
'use strict';

class UXComponent {
  constructor() {
    this.disabled = false;
    this.focused = false;
    this.hovered = false;
    this.active = false;
    this.hasIcon = false;
  }

  getClassName() {
    let classname = [];
    if (this.disabled) {
      classname.push('item-disabled');
    }
    if (this.focused) {
      classname.push('item-focused');
    }
    if (this.hovered) {
      classname.push('item-hovered');
    }
    if (this.active) {
      classname.push('item-active');
    }
    if (this.hasIcon) {
      classname.push('item-hasicon');
    }
    return classname.join(' ');
  }
}

```

untitled

```
function pow(number, power) {
  if (power === 0) return 1;
  if (power === 1) return number;
  if (power === 2) return number * number;
  if (power === 3) return number * number * number;
  if (power === 4) return number * number * number * number;
  if (power === 5) return number * number * number * number * number;
  if (power === 6) return number * number * number * number * number * number;
  if (power === 7) return number * number * number * number * number * number * number;
  if (power === 8) return number * number * number * number * number * number * number *
  if (power === 9) return number * number * number * number * number * number * number *
  if (power === 10) return number * number * number * number * number * number * number
  if (power === 11) return number * number * number * number * number * number * number
  if (power === 12) return number * number * number * number * number * number * number
  if (power === 13) return number * number * number * number * number * number * number
  if (power === 14) return number * number * number * number * number * number * number
  if (power === 15) return number * number * number * number * number * number * number
  if (power === 16) return number * number * number * number * number * number * number
  if (power === 17) return number * number * number * number * number * number * number
  if (power === 18) return number * number * number * number * number * number * number
  if (power === 19) return number * number * number * number * number * number * number
  if (power === 20) return number * number * number * number * number * number * number
  if (power === 21) return number * number * number * number * number * number * number
  if (power === 22) return number * number * number * number * number * number * number
  if (power === 23) return number * number * number * number * number * number * number
  if (power === 24) return number * number * number * number * number * number * number
  if (power === 25) return number * number * number * number * number * number * number
  if (power === 26) return number * number * number * number * number * number * number
  if (power === 27) return number * number * number * number * number * number * number
  if (power === 28) return number * number * number * number * number * number * number
  if (power === 29) return number * number * number * number * number * number * number
  if (power === 30) return number * number * number * number * number * number * number
  if (power === 31) return number * number * number * number * number * number * number
```

untitled

```
function pow(number, power) {
  if (power === 0) return 1;
  if (power === 1) return number;
  if (power === 2) return number * number;
  if (power === 3) return number * number * number;
  if (power === 4) return number * number * number * number;
  if (power === 5) return number * number * number * number * number;
  if (power === 6) return number * number * number * number * number * number;
  if (power === 7) return number * number * number * number * number * number * number;
  if (power === 8) return number * number * number * number * number * number * number *
  if (power === 9) return number * number * number * number * number * number * number *
  if (power === 10) return number * number * number * number * number * number * number
  if (power === 11) return number * number * number * number * number * number * number
  if (power === 12) return number * number * number * number * number * number * number
  if (power === 13) return number * number * number * number * number * number * number
  if (power === 14) return number * number * number * number * number * number * number
  if (power === 15) return number * number * number * number * number * number * number
  if (power === 16) return number * number * number * number * number * number * number
  if (power === 17) return number * number * number * number * number * number * number
  if (power === 18) return number * number * number * number * number * number * number
  if (power === 19) return number * number * number * number * number * number * number
  if (power === 20) return number * number * number * number * number * number * number
  if (power === 21) return number * number * number * number * number * number * number
  if (power === 22) return number * number * number * number * number * number * number
  if (power === 23) return number * number * number * number * number * number * number
  if (power === 24) return number * number * number * number * number * number * number
  if (power === 25) return number * number * number * number * number * number * number
  if (power === 26) return number * number * number * number * number * number * number
  if (power === 27) return number * number * number * number * number * number * number
  if (power === 28) return number * number * number * number * number * number * number
  if (power === 29) return number * number * number * number * number * number * number
  if (power === 30) return number * number * number * number * number * number * number
```

Битовые карты

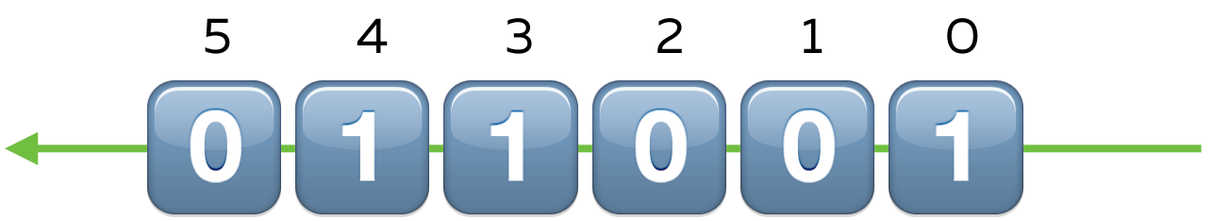
(bitmap) они же битовые массивы (bitset, bitarray)

Двоичная запись

В памяти компьютера все числа хранятся в двоичном виде — как последовательность нулей и единиц. Разряды читаются справа налево. Нумерация разрядов начинается с нуля. Единица в разряде означает что итоговое число содержит двойку
в степени номера разряда

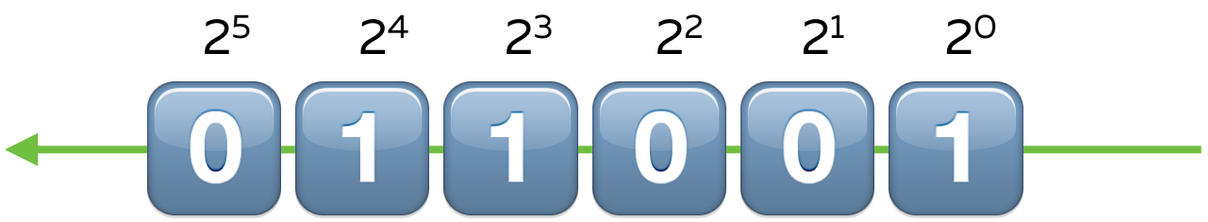
Двоичная запись

В памяти компьютера все числа хранятся в двоичном виде — как последовательность нулей и единиц. Разряды читаются справа налево. Нумерация разрядов начинается с нуля. Единица в разряде означает что итоговое число содержит двойку
в степени номера разряда



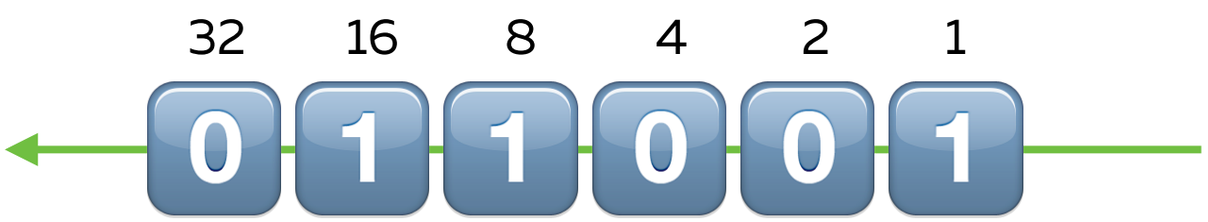
Двоичная запись

В памяти компьютера все числа хранятся в двоичном виде — как последовательность нулей и единиц. Разряды читаются справа налево. Нумерация разрядов начинается с нуля. Единица в разряде означает что итоговое число содержит двойку
в степени номера разряда



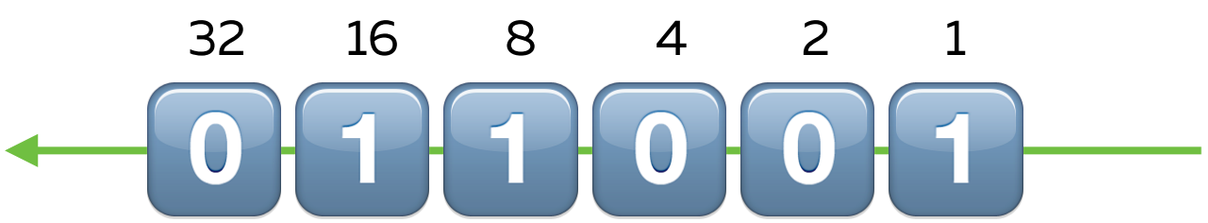
Двоичная запись

В памяти компьютера все числа хранятся в двоичном виде — как последовательность нулей и единиц. Разряды читаются справа налево. Нумерация разрядов начинается с нуля. Единица в разряде означает что итоговое число содержит двойку
в степени номера разряда



Двоичная запись

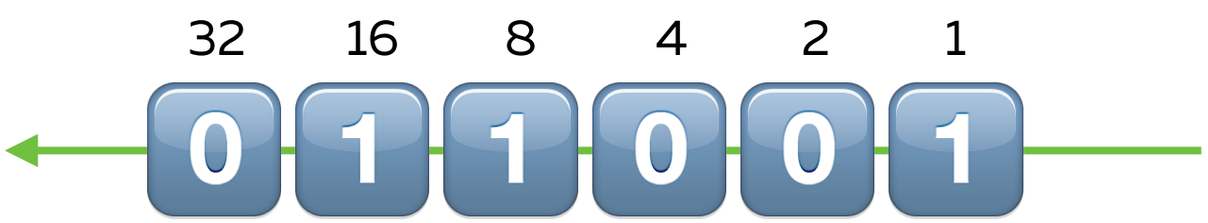
В памяти компьютера все числа хранятся в двоичном виде — как последовательность нулей и единиц. Разряды читаются справа налево. Нумерация разрядов начинается с нуля. Единица в разряде означает что итоговое число содержит двойку
в степени номера разряда



0 + 16 + 8 + 0 + 0 + 1

Двоичная запись

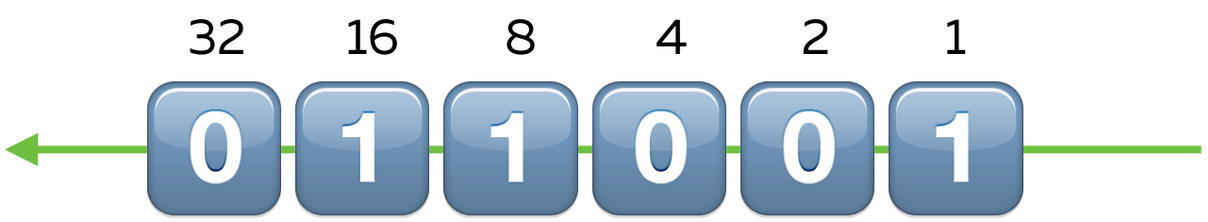
В памяти компьютера все числа хранятся в двоичном виде — как последовательность нулей и единиц. Разряды читаются справа налево. Нумерация разрядов начинается с нуля. Единица в разряде означает что итоговое число содержит двойку
в степени номера разряда



0 + 16 + 8 + 0 + 0 + 1 =25

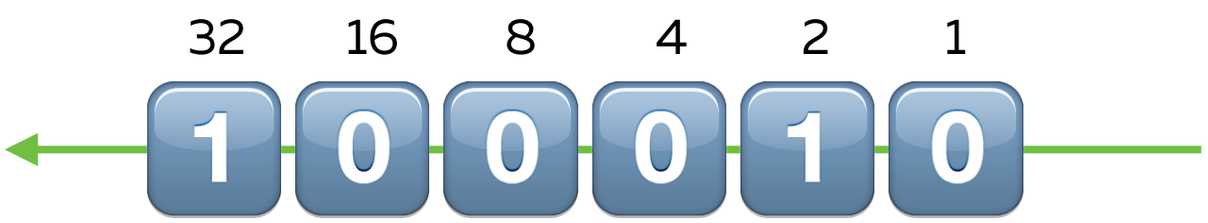
Двоичная запись

В памяти компьютера все числа хранятся в двоичном виде — как последовательность нулей и единиц. Разряды читаются справа налево. Нумерация разрядов начинается с нуля. Единица в разряде означает что итоговое число содержит двойку
в степени номера разряда



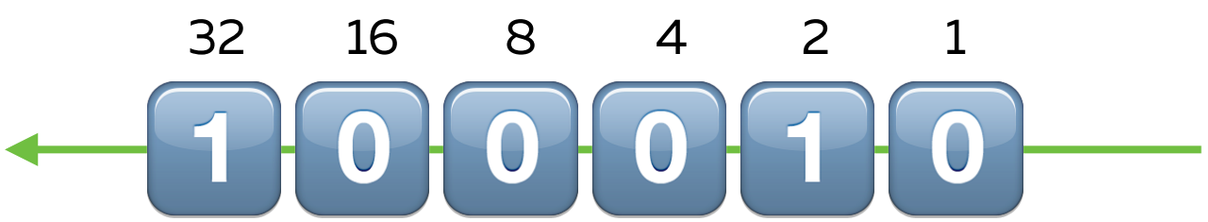
Двоичная запись

В памяти компьютера все числа хранятся в двоичном виде — как последовательность нулей и единиц. Разряды читаются справа налево. Нумерация разрядов начинается с нуля. Единица в разряде означает что итоговое число содержит двойку
в степени номера разряда



Двоичная запись

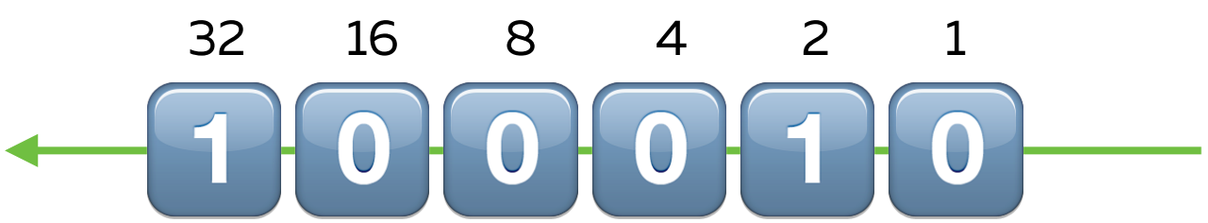
В памяти компьютера все числа хранятся в двоичном виде — как последовательность нулей и единиц. Разряды читаются справа налево. Нумерация разрядов начинается с нуля. Единица в разряде означает что итоговое число содержит двойку
в степени номера разряда



32 + 0 + 0 + 0 + 2 + 0

Двоичная запись

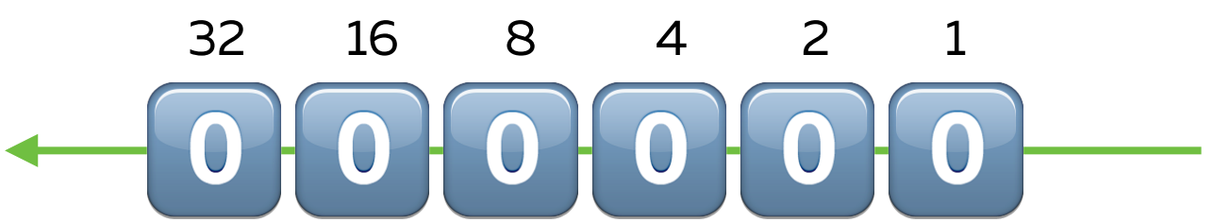
В памяти компьютера все числа хранятся в двоичном виде — как последовательность нулей и единиц. Разряды читаются справа налево. Нумерация разрядов начинается с нуля. Единица в разряде означает что итоговое число содержит двойку
в степени номера разряда



32 + 0 + 0 + 0 + 2 + 0 =34

Двоичная запись

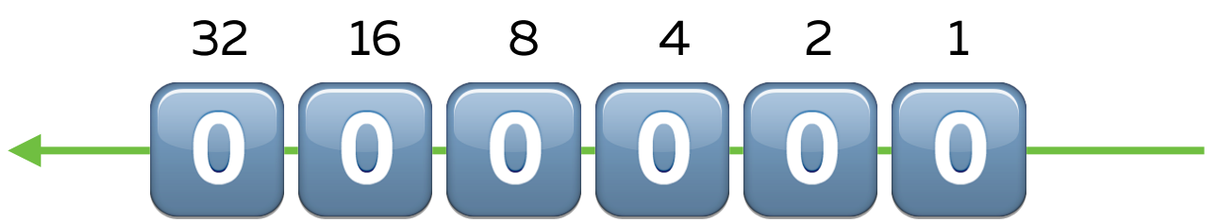
В памяти компьютера все числа хранятся в двоичном виде — как последовательность нулей и единиц. Разряды читаются справа налево. Нумерация разрядов начинается с нуля. Единица в разряде означает что итоговое число содержит двойку
в степени номера разряда



19= 0 + 0 + 0 + 0 + 0 + 0

Двоичная запись

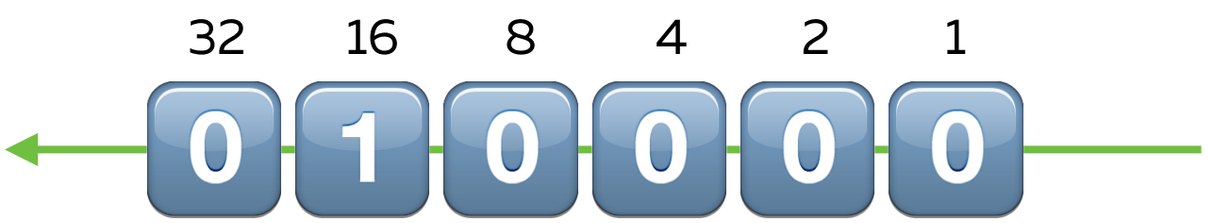
В памяти компьютера все числа хранятся в двоичном виде — как последовательность нулей и единиц. Разряды читаются справа налево. Нумерация разрядов начинается с нуля. Единица в разряде означает что итоговое число содержит двойку
в степени номера разряда



19= 0 + 16 + 0 + 0 + 0 + 0

Двоичная запись

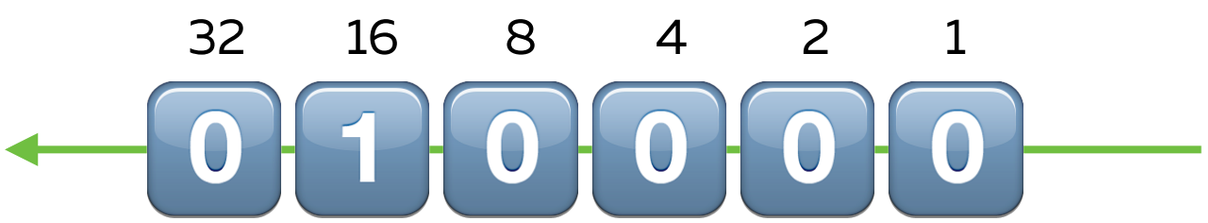
В памяти компьютера все числа хранятся в двоичном виде — как последовательность нулей и единиц. Разряды читаются справа налево. Нумерация разрядов начинается с нуля. Единица в разряде означает что итоговое число содержит двойку
в степени номера разряда



19= 0 + 16 + 0 + 0 + 0 + 0

Двоичная запись

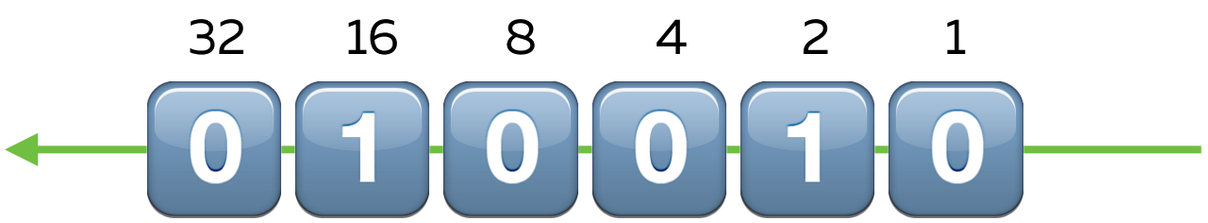
В памяти компьютера все числа хранятся в двоичном виде — как последовательность нулей и единиц. Разряды читаются справа налево. Нумерация разрядов начинается с нуля. Единица в разряде означает что итоговое число содержит двойку
в степени номера разряда



19= 0 + 16 + 0 + 0 + 2 + 0

Двоичная запись

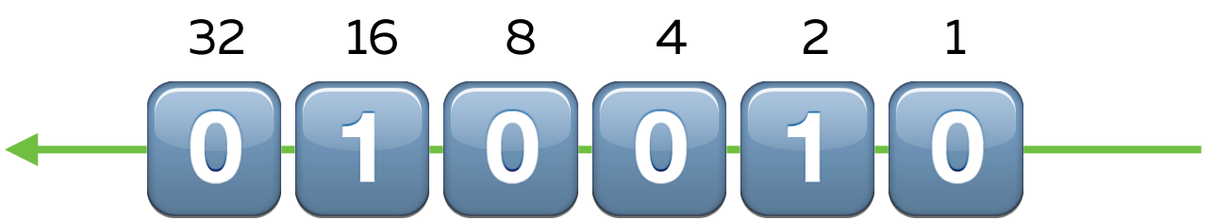
В памяти компьютера все числа хранятся в двоичном виде — как последовательность нулей и единиц. Разряды читаются справа налево. Нумерация разрядов начинается с нуля. Единица в разряде означает что итоговое число содержит двойку
в степени номера разряда



19= 0 + 16 + 0 + 0 + 2 + 0

Двоичная запись

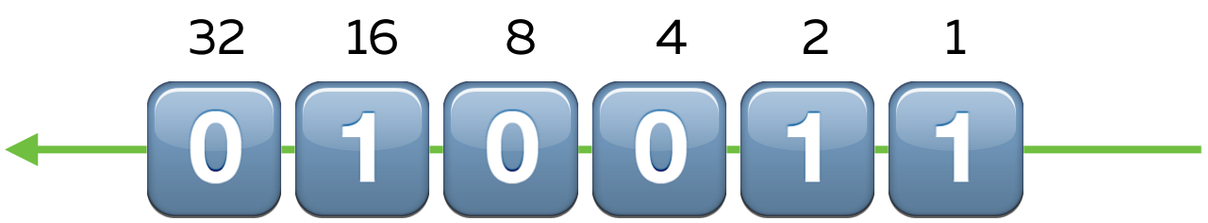
В памяти компьютера все числа хранятся в двоичном виде — как последовательность нулей и единиц. Разряды читаются справа налево. Нумерация разрядов начинается с нуля. Единица в разряде означает что итоговое число содержит двойку
в степени номера разряда



19= 0 + 16 + 0 + 0 + 2 + 1

Двоичная запись

В памяти компьютера все числа хранятся в двоичном виде — как последовательность нулей и единиц. Разряды читаются справа налево. Нумерация разрядов начинается с нуля. Единица в разряде означает что итоговое число содержит двойку
в степени номера разряда



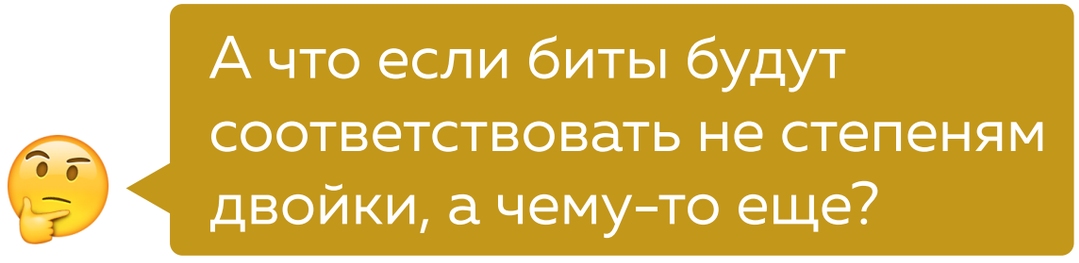
19= 0 + 16 + 0 + 0 + 2 + 1

Двоичная запись

В памяти компьютера все числа хранятся в двоичном виде — как последовательность нулей и единиц. Разряды читаются справа налево. Нумерация разрядов начинается с нуля. Единица в разряде означает что итоговое число содержит двойку
в степени номера разряда

Двоичная запись

В памяти компьютера все числа хранятся в двоичном виде — как последовательность нулей и единиц. Разряды читаются справа налево. Нумерация разрядов начинается с нуля. Единица в разряде означает что итоговое число содержит двойку
в степени номера разряда



Битовые карты

битовой картой называется последовательность бит (нулей и единиц). В битовых картах может храниться не только число, но и любой другой набор данных

Битовые карты

битовой картой называется последовательность бит (нулей и единиц). В битовых картах может храниться не только число, но и любой другой набор данных

- IP-адрес
  `192.168.1.1/255.255.255.0`

Битовые карты

битовой картой называется последовательность бит (нулей и единиц). В битовых картах может храниться не только число, но и любой другой набор данных

- IP-адрес
  `192.168.1.1/255.255.255.0`

- цвет
  `#FACE8D`

Битовые карты

битовой картой называется последовательность бит (нулей и единиц). В битовых картах может храниться не только число, но и любой другой набор данных

- IP-адрес
  `192.168.1.1/255.255.255.0`

- цвет
  `#FACE8D`

- права пользователя
  `chmod -R 777 .`

Битовые карты

битовой картой называется последовательность бит (нулей и единиц). В битовых картах может храниться не только число, но и любой другой набор данных

- IP-адрес
  `192.168.1.1/255.255.255.0`

- цвет
  `#FACE8D`

- права пользователя
  `chmod -R 777 .`

- *карта уровня в игре* или *пиксели изображения* (bitmap images, хранятся в двумерных массивах)

Битовые карты

битовой картой называется последовательность бит (нулей и единиц). В битовых картах может храниться не только число, но и любой другой набор данных

- IP-адрес
  `192.168.1.1/255.255.255.0`

- цвет
  `#FACE8D`

- права пользователя
  `chmod -R 777 .`

- *карта уровня в игре* или *пиксели изображения* (bitmap images, хранятся в двумерных массивах)

- состояние UX-компоненты

component.js — /Users/igoralexeenko/dev/ha/demo.js

component.js

```
1   'use strict';
2
3   class UXComponent {
4     static get State() {
5
6     }
7
8     constructor() {
9
10    }
11  }
12
```

Line 0 | File 0 | Project 0 | No Issues | component.js | 6:4 | LF | UTF-8 | JavaScript | master | +12

component.js — /Users/igoralexeenko/dev/ha/demo.js

component.js

```
1   'use strict';
2
3   class UXComponent {
4     static get State() {
5       return {
6
7       };
8     }
9
10    constructor() {
11
12    }
13  }
14
```

Line 0 | File 0 | Project 0 | No Issues | component.js | 7:7 | LF | UTF-8 | JavaScript | master | +14

component.js — /Users/igoralexeenko/dev/ha/demo.js

component.js

```
1   'use strict';
2
3   class UXComponent {
4     static get State() {
5       return {
6         DISABLED: 0x01
7       };
8     }
9
10    constructor() {
11
12    }
13  }
14
```

Line 0 | File 0 | Project 0 | No Issues | component.js | 6:21 | LF | UTF-8 | JavaScript | master | +14

component.js — /Users/igoralexeenko/dev/ha/demo.js

component.js

```
1   'use strict';
2
3   class UXComponent {
4     static get State() {
5       return {
6         DISABLED: 0x01,
7         FOCUSED: 0x02
8       };
9     }
10
11    constructor() {
12
13    }
14  }
15
```

Line 0 File 0 Project 0 ✔ No Issues component.js 7:20 LF UTF-8 JavaScript master +15

component.js — /Users/igoralexeenko/dev/ha/demo.js

component.js

```
1   'use strict';
2
3   class UXComponent {
4     static get State() {
5       return {
6         DISABLED: 0x01,
7         FOCUSED: 0x02,
8         HOVERED: 0x04
9       };
10    }
11
12    constructor() {
13
14    }
15  }
16
```

Line 0 File 0 Project 0 ✔ No Issues component.js 8:20 LF UTF-8 JavaScript master +16

component.js — /Users/igoralexeenko/dev/ha/demo.js

component.js

```
1   'use strict';
2
3   class UXComponent {
4     static get State() {
5       return {
6         DISABLED: 0x01,
7         FOCUSED: 0x02,
8         HOVERED: 0x04,
9         ACTIVE: 0x08
10      };
11    }
12
13    constructor() {
14
15    }
16  }
17
```

Line 0 File 0 Project 0 ✔ No Issues component.js 9:19 LF UTF-8 JavaScript master +17

component.js — /Users/igoralexeenko/dev/ha/demo.js

component.js

```
'use strict';

class UXComponent {
  static get State() {
    return {
      DISABLED: 0x01,
      FOCUSED: 0x02,
      HOVERED: 0x04,
      ACTIVE: 0x08,
      HAS_ICON: 0x10
    };
  }

  constructor() {

  }

}
```

Line 0 File 0 Project 0 ✔ No Issues component.js 10:21 LF UTF-8 JavaScript master +18

Битовые операции

логические операции над цепочками битов, в которых биты выступают как значения true или false. Операции производятся побитово над соответствующими битами. При несовпадении разрядности, к числу с меньшим количеством разрядов добавляются нули

Битовые операции

логические операции над цепочками битов, в которых биты выступают как значения true или false. Операции производятся побитово над соответствующими битами. При несовпадении разрядности, к числу с меньшим количеством разрядов добавляются нули

- отицание

```
NOT 0000     NOT 1111     NOT 0010     NOT 0101
  = 1111       = 0000       = 1101       = 1010
```

Битовые операции

логические операции над цепочками битов, в которых биты выступают как значения true или false. Операции производятся побитово над соответствующими битами. При несовпадении разрядности, к числу с меньшим количеством разрядов добавляются нули

- отицание

```
NOT 0000     NOT 1111     NOT 0010     NOT 0101
  = 1111       = 0000       = 1101       = 1010
```

- побитовое «или»

```
    0001         0000         0011         1011
 OR 0001      OR 0000      OR 0010      OR 1110
  = 0001       = 0000       = 0011       = 1111
```


Битовые операции

логические операции над цепочками битов, в которых биты выступают как значения true или false. Операции производятся побитово над соответствующими битами. При несовпадении разрядности, к числу с меньшим количеством разрядов добавляются нули

- отицание

```
NOT 0000     NOT 1111     NOT 0010     NOT 0101
  = 1111       = 0000       = 1101       = 1010
```

- побитовое «или»

```
    0001         0000         0011         1011
 OR 0001      OR 0000      OR 0010      OR 1110
  = 0001       = 0000       = 0011       = 1111
```

- побитовое «и»

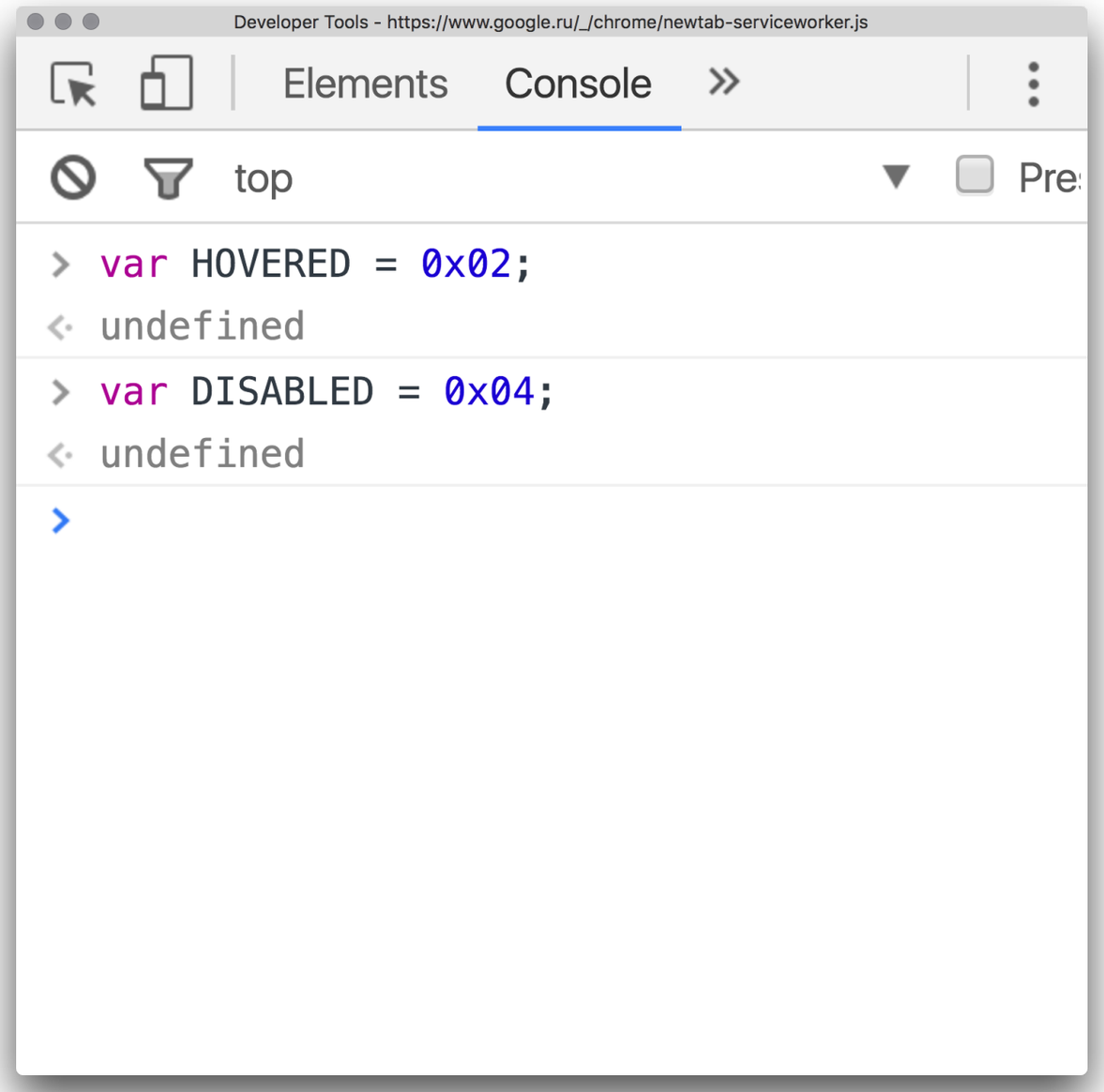
```
    0001         0000         0011         1011
AND 0001     AND 0000     AND 0010     AND 1110
  = 0001       = 0000       = 0010       = 1010
```

Побитовое «или»

складывает переданные значения и сохраняет включенные биты обоих операторов. Идеально подходит для сложения нескольких состояний

Побитовое «или»

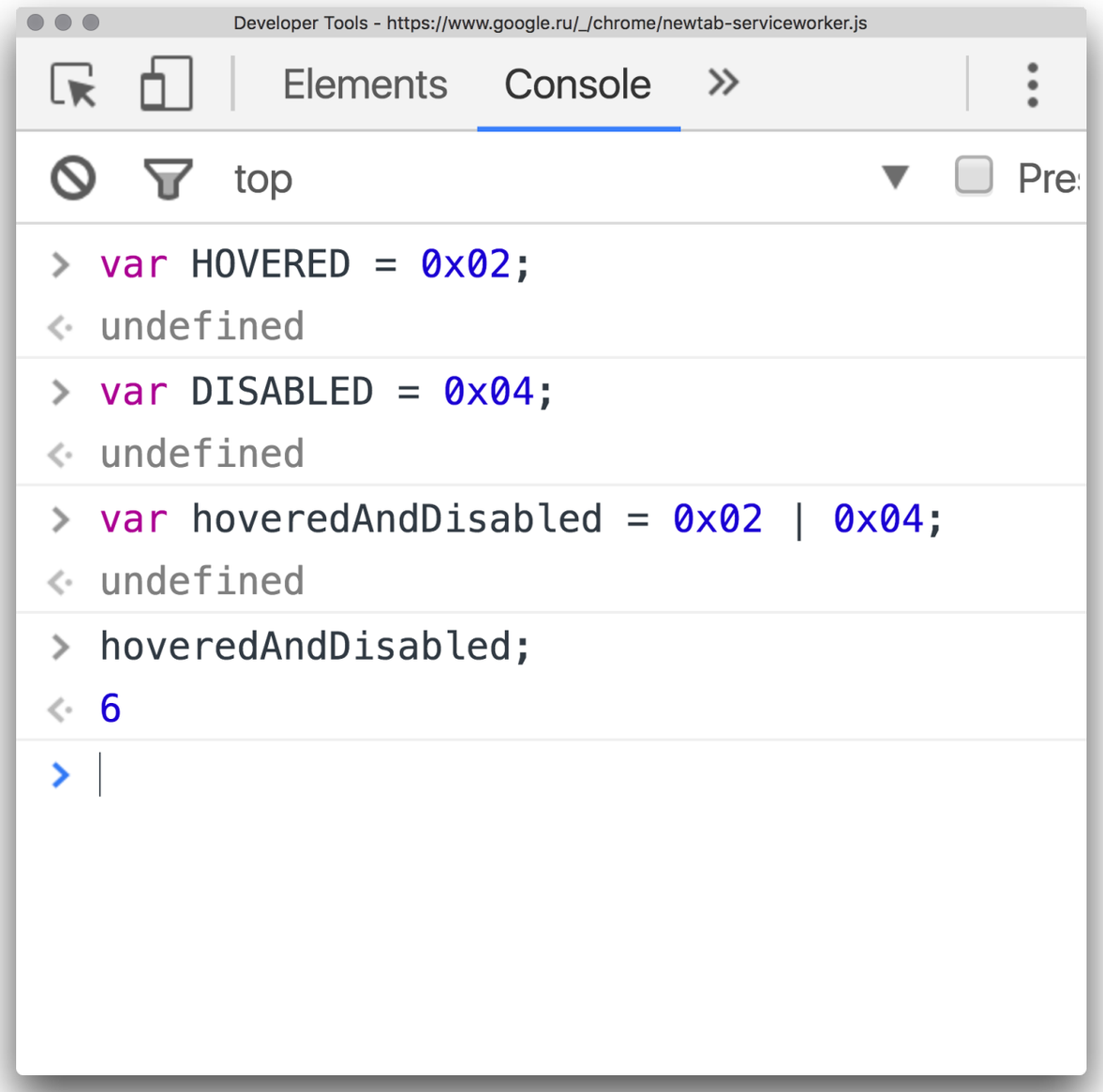
складывает переданные значения и сохраняет включенные биты обоих операторов. Идеально подходит для сложения нескольких состояний




```
010
OR 100
```

Побитовое «или»

складывает переданные значения и сохраняет включенные биты обоих операторов. Идеально подходит для сложения нескольких состояний



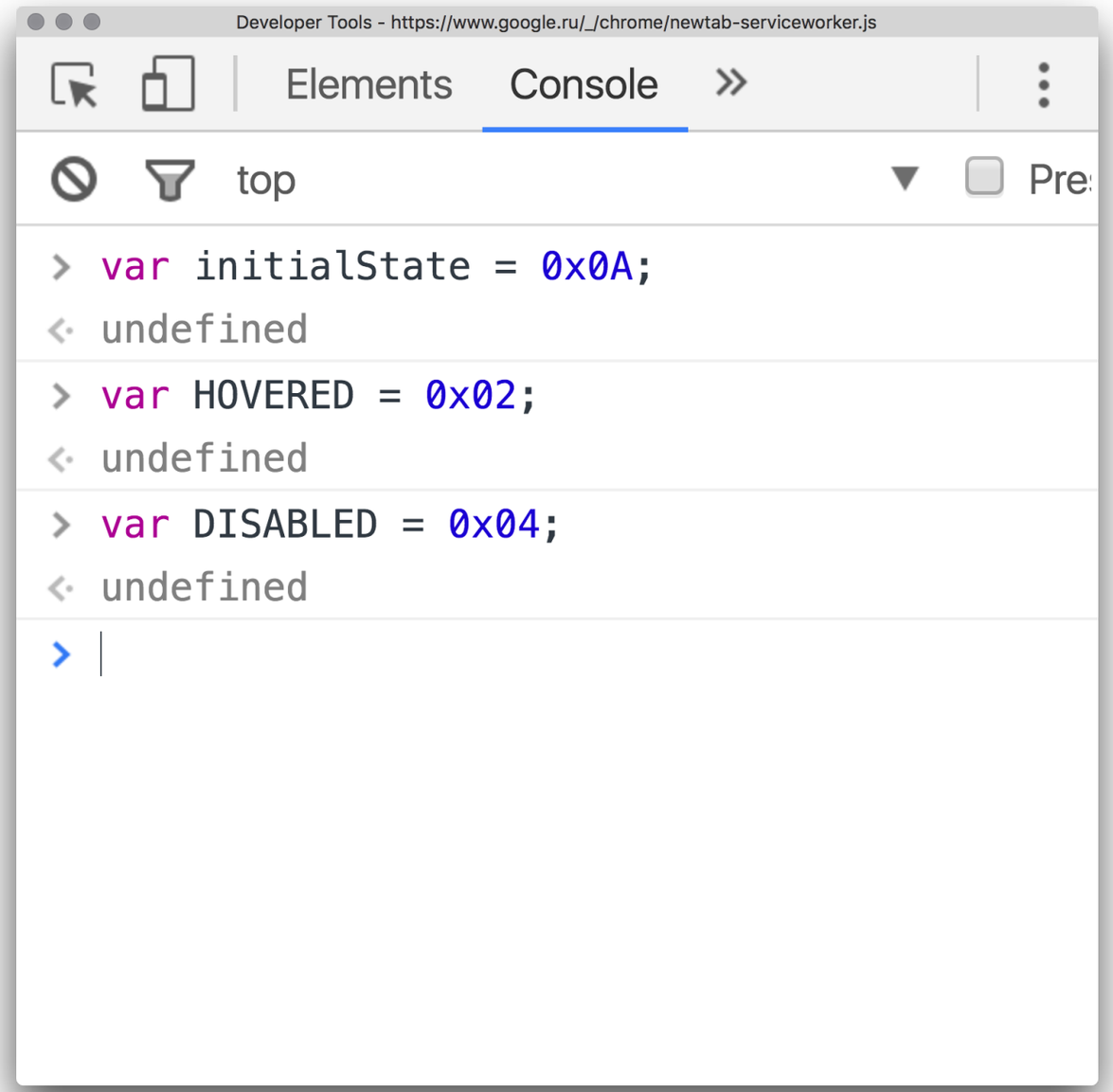

```
  010
OR 100
 = 110
```

Побитовое «и»

сохраняет только те биты, которые были включены в обоих операндах. Идеально подходит для проверки состояния

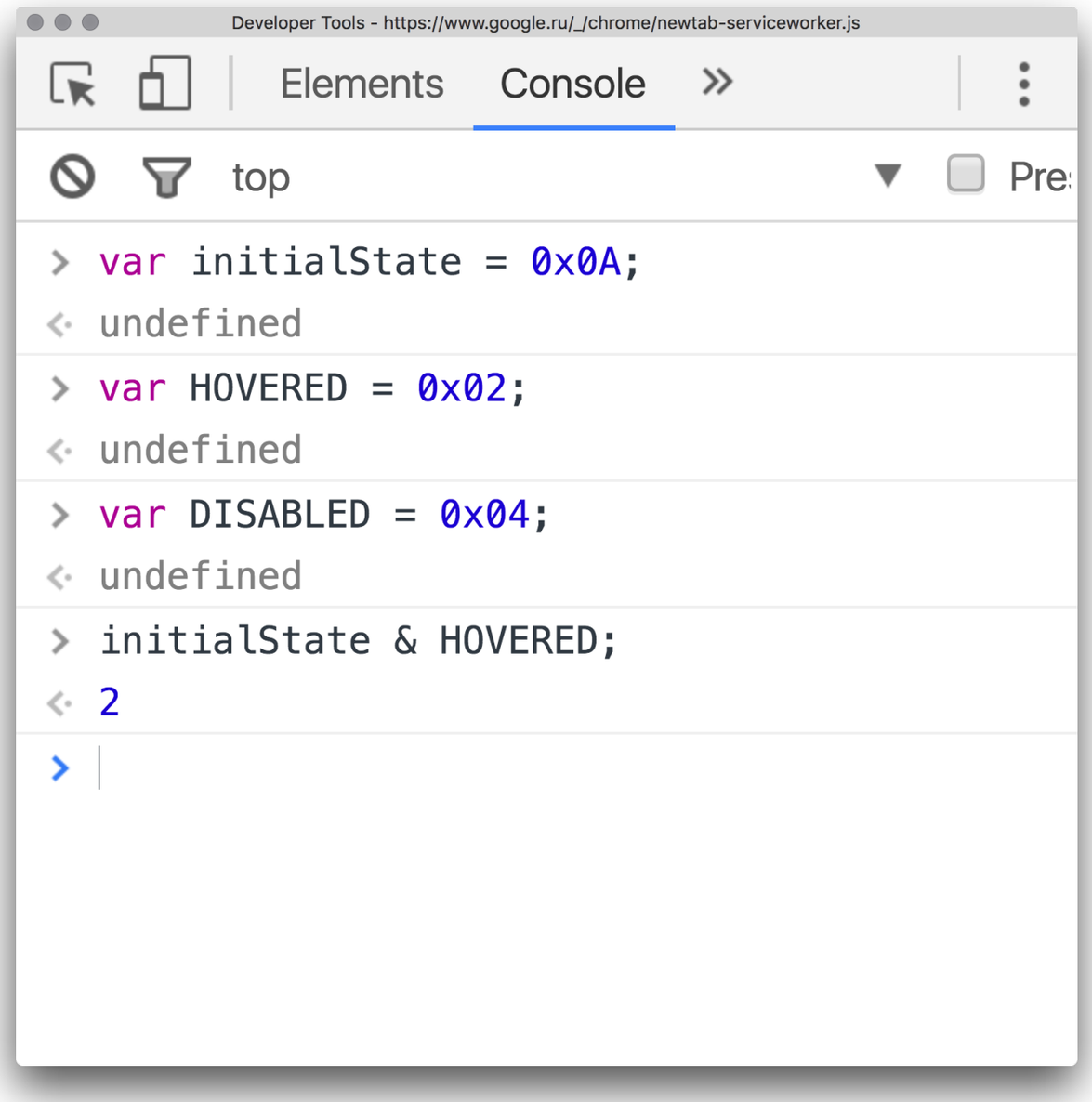
Побитовое «и»

сохраняет только те биты, которые были включены в обоих операндах. Идеально подходит для проверки состояния



Побитовое «и»

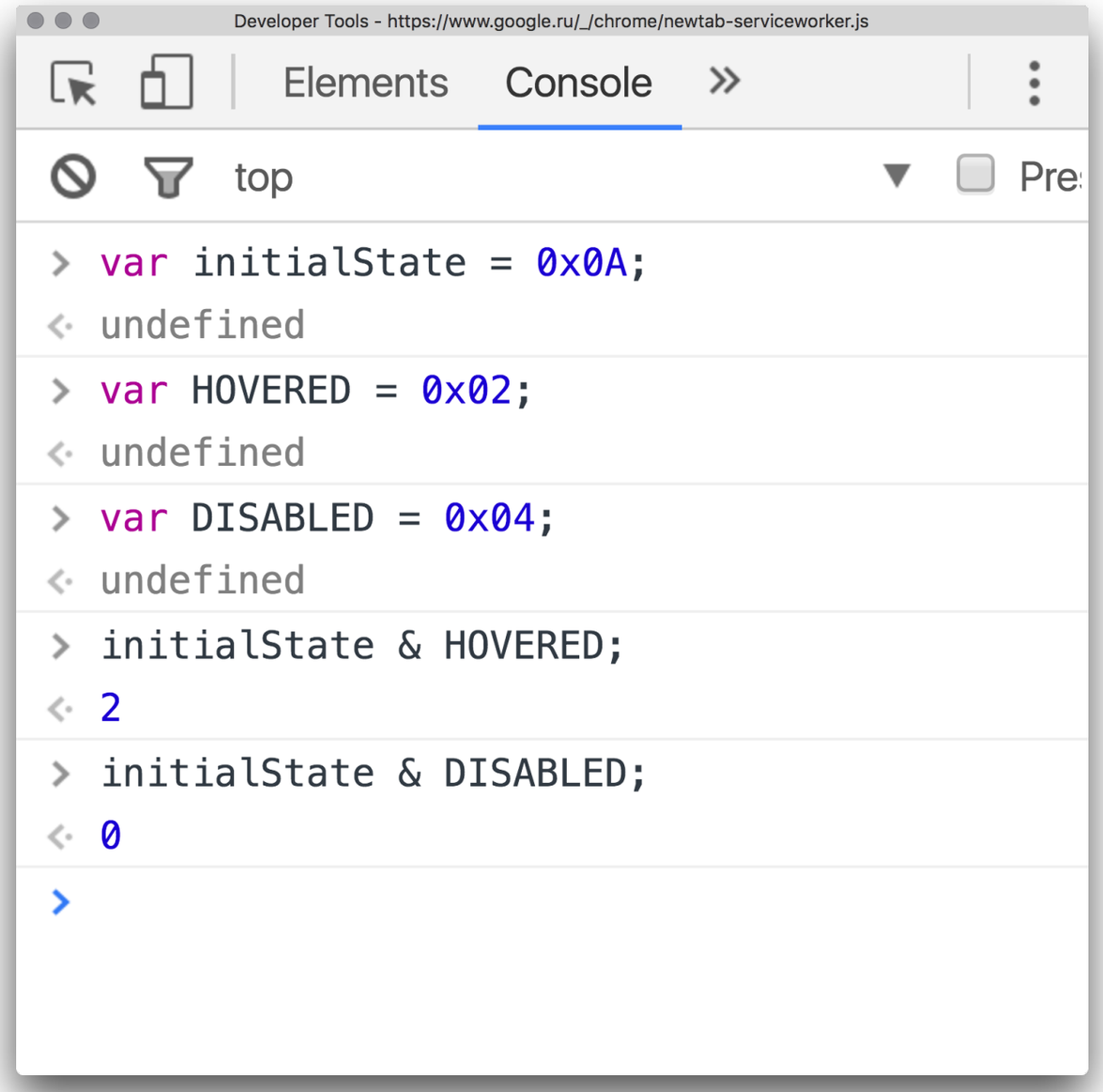
сохраняет только те биты, которые были включены в обоих операндах. Идеально подходит для проверки состояния




```
1010
AND 0010
  = 0010
```

Побитовое «и»

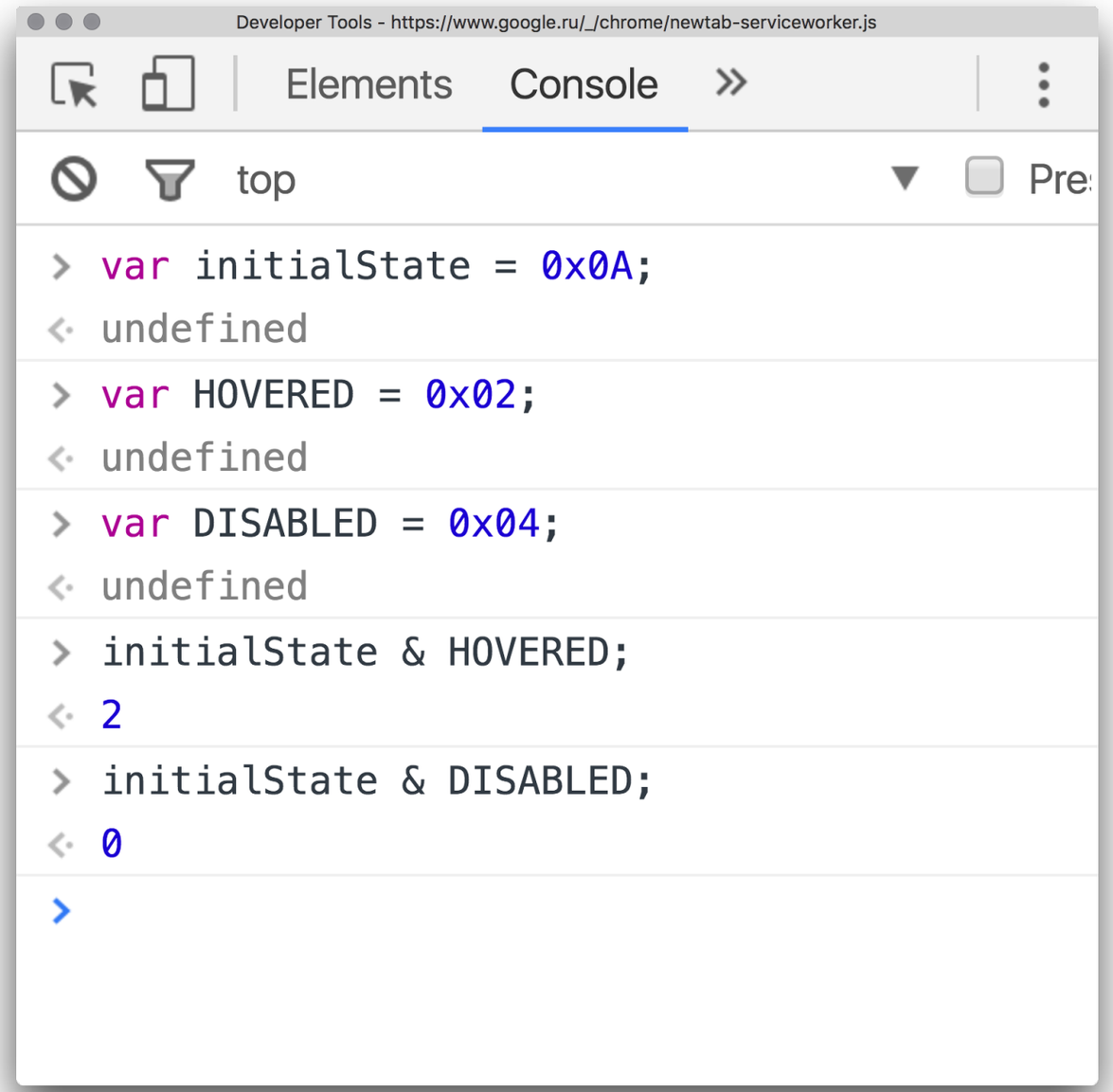
сохраняет только те биты, которые были включены в обоих операндах. Идеально подходит для проверки состояния




```
1010
AND 0100
  = 0000
```

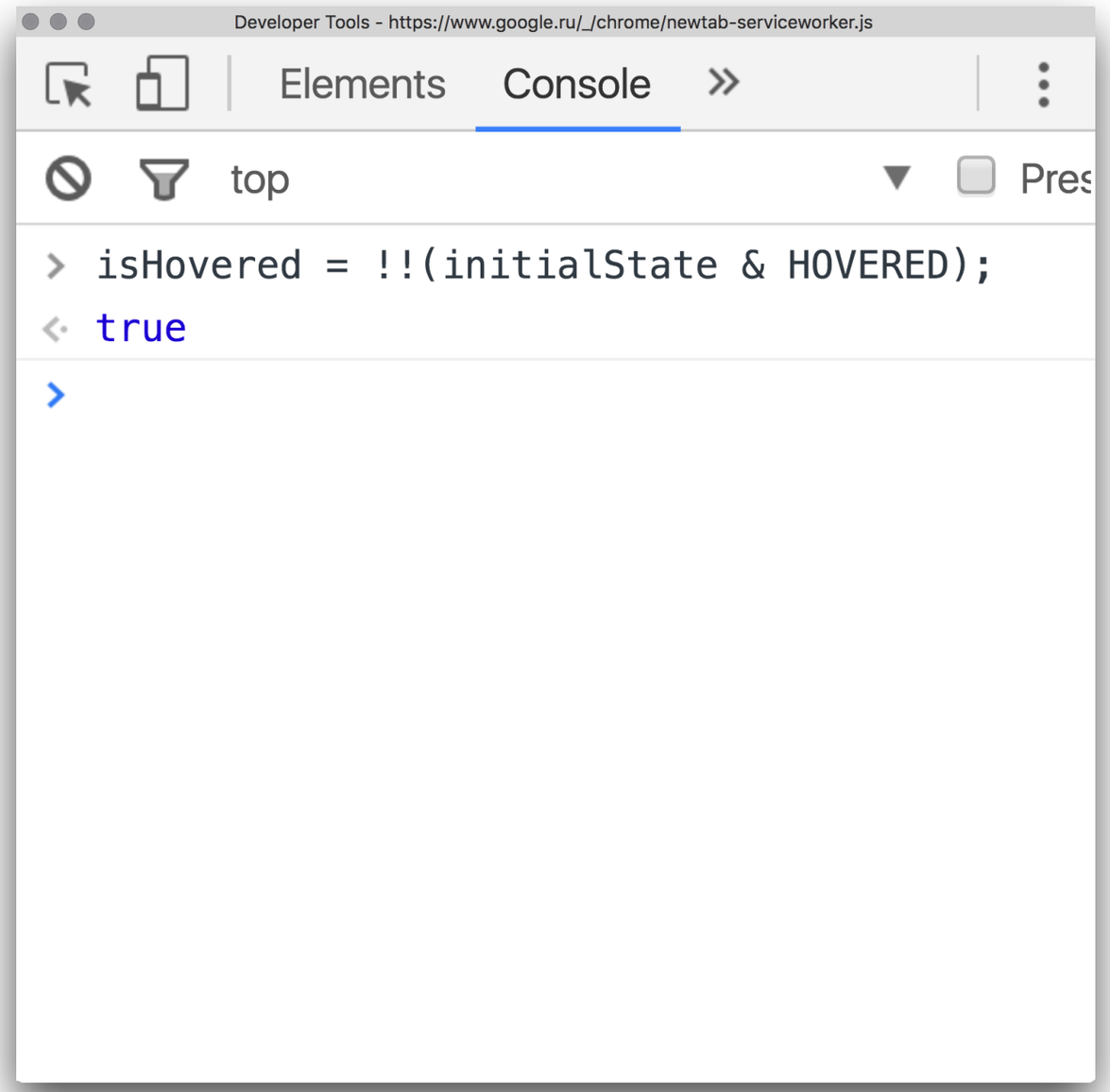
Побитовое «и»

сохраняет только те биты, которые были включены в обоих операндах. Идеально подходит для проверки состояния



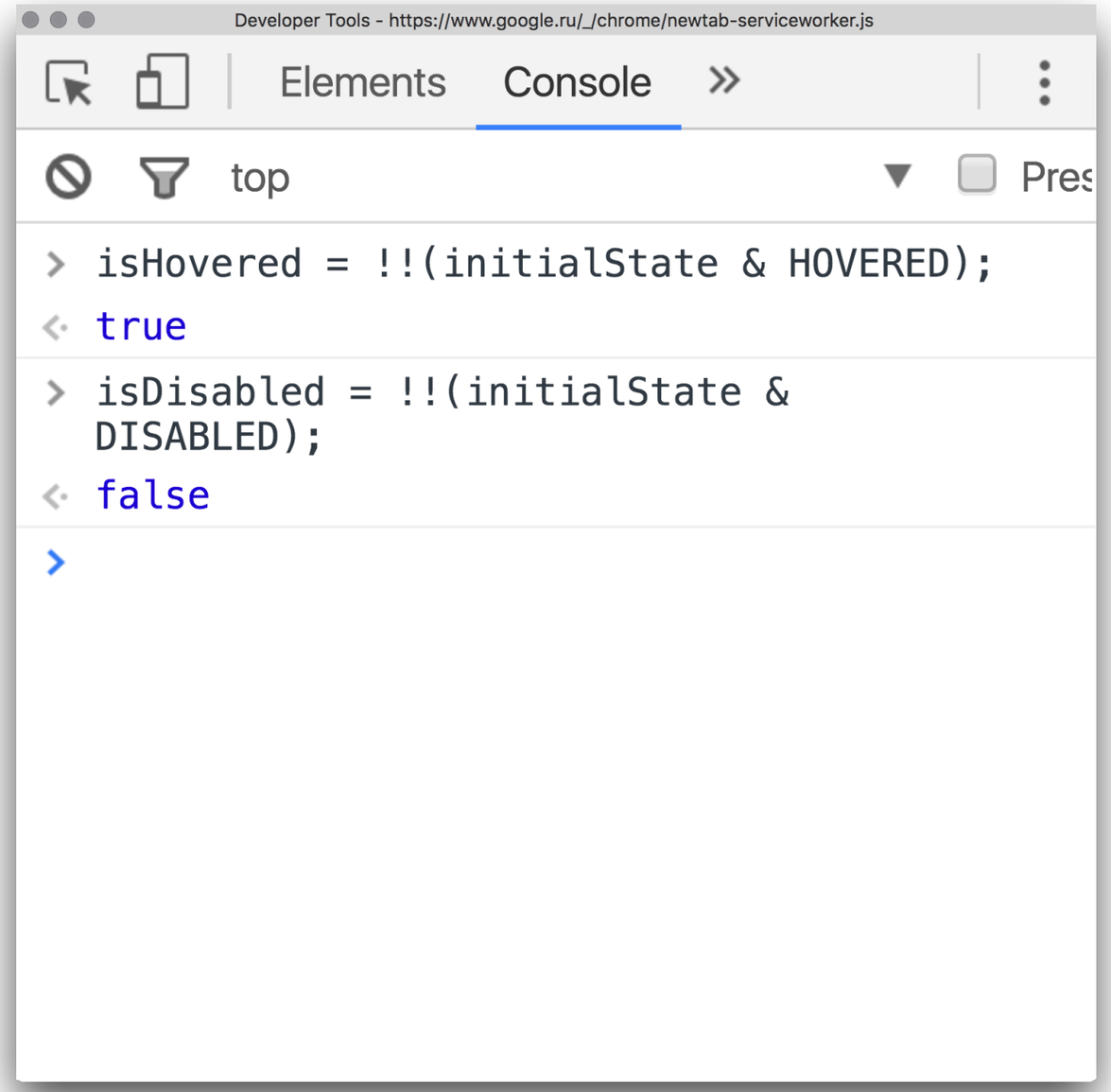
Побитовое «и»

сохраняет только те биты, которые были включены в обоих операндах. Идеально подходит для проверки состояния



Побитовое «и»

сохраняет только те биты, которые были включены в обоих операндах. Идеально подходит для проверки состояния



component.js — /Users/igoralexeenko/dev/ha/demo.js

component.js

```
'use strict';

class UXComponent {
  static get State() {
    return {
      DISABLED: 0x01,
      FOCUSED: 0x02,
      HOVERED: 0x04,
      ACTIVE: 0x08,
      HAS_ICON: 0x10
    };
  }

  static get StateClassName() {

  }

  constructor() {
    this.state = UXComponent.State.DISABLED | UXComponent.State.HAS_ICON;
  }

  getClassName() {

  }
}

```

Line 0 File 0 Project 0 ✔ No Issues component.js 16:4 LF UTF-8 JavaScript master +26

component.js — /Users/igoralexeenko/dev/ha/demo.js

component.js

```
'use strict';

class UXComponent {
  static get State() {
    return {
      DISABLED: 0x01,
      FOCUSED: 0x02,
      HOVERED: 0x04,
      ACTIVE: 0x08,
      HAS_ICON: 0x10
    };
  }

  static get StateClassName() {
    return new Map([
      [UXComponent.State.DISABLED, 'item-disabled'],
    ]);
  }

  constructor() {
    this.state = UXComponent.State.DISABLED | UXComponent.State.HAS_ICON;
  }

  getClassName() {

  }
}

```

Line 0 File 0 Project 0 No Issues component.js 16:53 LF UTF-8 JavaScript master +28

component.js — /Users/igoralexeenko/dev/ha/demo.js

component.js

```
'use strict';

class UXComponent {
  static get State() {
    return {
      DISABLED: 0x01,
      FOCUSED: 0x02,
      HOVERED: 0x04,
      ACTIVE: 0x08,
      HAS_ICON: 0x10
    };
  }

  static get StateClassName() {
    return new Map([
      [UXComponent.State.DISABLED, 'item-disabled'],
      [UXComponent.State.FOCUCED, 'item-focused'],
      [UXComponent.State.HOVERED, 'item-hovered'],
    ]);
  }

  constructor() {
    this.state = UXComponent.State.DISABLED | UXComponent.State.HAS_ICON;
  }

  getClassName() {

  }
}

```

Line 0 File 0 Project 0 No Issues component.js 18:48 LF UTF-8 JavaScript master +30

component.js — /Users/igoralexeenko/dev/ha/demo.js

component.js

```
'use strict';

class UXComponent {
  static get State() {
    return {
      DISABLED: 0x01,
      FOCUSED: 0x02,
      HOVERED: 0x04,
      ACTIVE: 0x08,
      HAS_ICON: 0x10
    };
  }

  static get StateClassName() {
    return new Map([
      [UXComponent.State.DISABLED, 'item-disabled'],
      [UXComponent.State.FOCUCED, 'item-focused'],
      [UXComponent.State.HOVERED, 'item-hovered'],
      [UXComponent.State.ACTIVE, 'item-active'],
    ]);
  }

  constructor() {
    this.state = UXComponent.State.DISABLED | UXComponent.State.HAS_ICON;
  }

  getClassName() {

  }
}

```

component.js — /Users/igoralexeenko/dev/ha/demo.js

component.js

```
'use strict';

class UXComponent {
  static get State() {
    return {
      DISABLED: 0x01,
      FOCUSED: 0x02,
      HOVERED: 0x04,
      ACTIVE: 0x08,
      HAS_ICON: 0x10
    };
  }

  static get StateClassName() {
    return new Map([
      [UXComponent.State.DISABLED, 'item-disabled'],
      [UXComponent.State.FOCUCED, 'item-focused'],
      [UXComponent.State.HOVERED, 'item-hovered'],
      [UXComponent.State.ACTIVE, 'item-active'],
      [UXComponent.State.HAS_ICON, 'item-hasicon']
    ]);
  }

  constructor() {
    this.state = UXComponent.State.DISABLED | UXComponent.State.HAS_ICON;
  }

  getClassName() {

  }
}

```

Line 0 File 0 Project 0 No Issues component.js 20:51 LF UTF-8 JavaScript master +32

component.js

```
'use strict';

class UXComponent {
  static get State() {
    return {
      DISABLED: 0x01,
      FOCUSED: 0x02,
      HOVERED: 0x04,
      ACTIVE: 0x08,
      HAS_ICON: 0x10
    };
  }

  static get StateClassName() {
    return new Map([
      [UXComponent.State.DISABLED, 'item-disabled'],
      [UXComponent.State.FOCUCED, 'item-focused'],
      [UXComponent.State.HOVERED, 'item-hovered'],
      [UXComponent.State.ACTIVE, 'item-active'],
      [UXComponent.State.HAS_ICON, 'item-hasicon']
    ]);
  }

  constructor() {
    this.state = UXComponent.State.DISABLED | UXComponent.State.HAS_ICON;
  }

  getClassName() {
    let classnames = [];

  }
}
```

component.js

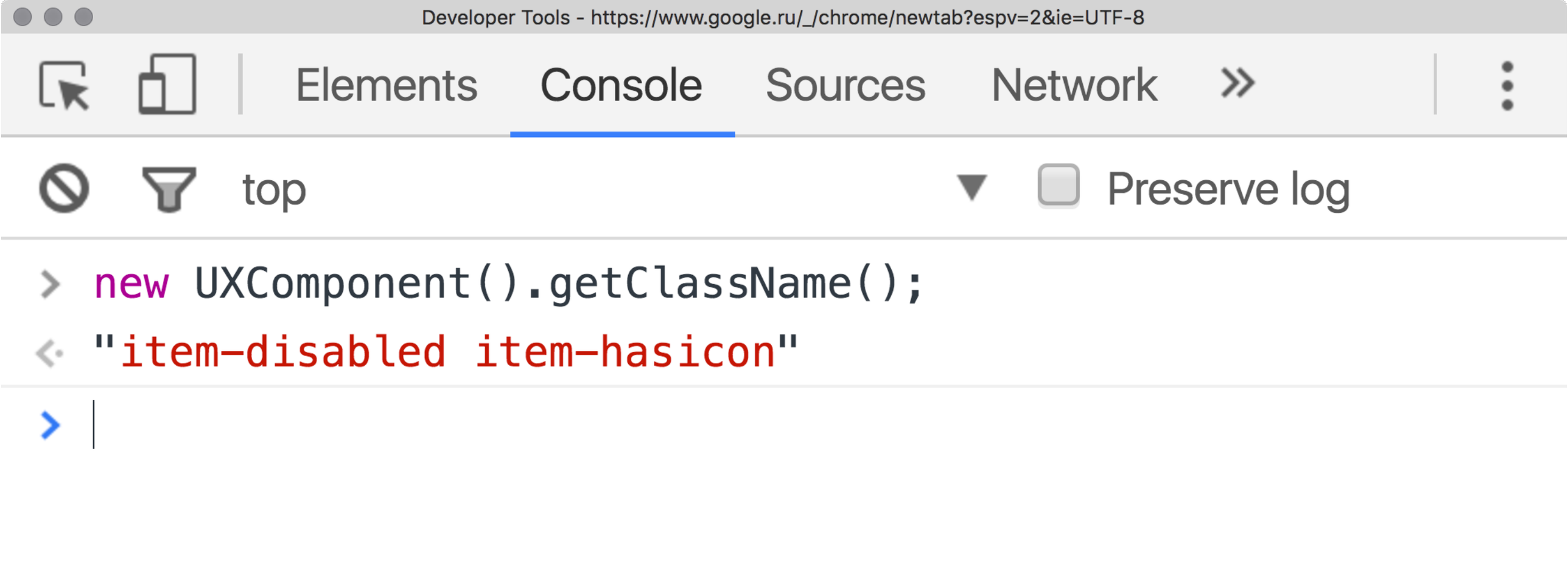
```
4       static get State() {
5         return {
6           DISABLED: 0x01,
7           FOCUSED: 0x02,
8           HOVERED: 0x04,
9           ACTIVE: 0x08,
10          HAS_ICON: 0x10
11        };
12      }
13
14      static get StateClassName() {
15        return new Map([
16          [UXComponent.State.DISABLED, 'item-disabled'],
17          [UXComponent.State.FOCUCED, 'item-focused'],
18          [UXComponent.State.HOVERED, 'item-hovered'],
19          [UXComponent.State.ACTIVE, 'item-active'],
20          [UXComponent.State.HAS_ICON, 'item-hasicon']
21        ]);
22      }
23
24      constructor() {
25        this.state = UXComponent.State.DISABLED | UXComponent.State.HAS_ICON;
26      }
27
28      getClassName() {
29        let classnames = [];
30        UXComponent.StateClassName.forEach((classname, state) => {
31
32        });
33        return classnames.join(' ');
34      }
35    }
36
```

component.js

```
  static get State() {
    return {
      DISABLED: 0x01,
      FOCUSED: 0x02,
      HOVERED: 0x04,
      ACTIVE: 0x08,
      HAS_ICON: 0x10
    };
  }

  static get StateClassName() {
    return new Map([
      [UXComponent.State.DISABLED, 'item-disabled'],
      [UXComponent.State.FOCUCED, 'item-focused'],
      [UXComponent.State.HOVERED, 'item-hovered'],
      [UXComponent.State.ACTIVE, 'item-active'],
      [UXComponent.State.HAS_ICON, 'item-hasicon']
    ]);
  }

  constructor() {
    this.state = UXComponent.State.DISABLED | UXComponent.State.HAS_ICON;
  }

  getClassName() {
    let classnames = [];
    UXComponent.StateClassName.forEach((classname, state) => {
      if (!!(this.state & state)) {

      }
    });
    return classnames.join(' ');
  }
```

component.js

```
  static get State() {
    return {
      DISABLED: 0x01,
      FOCUSED: 0x02,
      HOVERED: 0x04,
      ACTIVE: 0x08,
      HAS_ICON: 0x10
    };
  }

  static get StateClassName() {
    return new Map([
      [UXComponent.State.DISABLED, 'item-disabled'],
      [UXComponent.State.FOCUCED, 'item-focused'],
      [UXComponent.State.HOVERED, 'item-hovered'],
      [UXComponent.State.ACTIVE, 'item-active'],
      [UXComponent.State.HAS_ICON, 'item-hasicon']
    ]);
  }

  constructor() {
    this.state = UXComponent.State.DISABLED | UXComponent.State.HAS_ICON;
  }

  getClassName() {
    let classnames = [];
    UXComponent.StateClassName.forEach((classname, state) => {
      if (!!(this.state & state)) {
        classnames.push(classname);
      }
    });
    return classnames.join(' ');
  }
```


Developer Tools - https://www.google.ru/_/chrome/newtab?espv=2&ie=UTF-8
Elements
Console
Sources
Network
top
Preserve log
new UXComponent().getClassName();
"item-disabled item-hasicon"

component.js

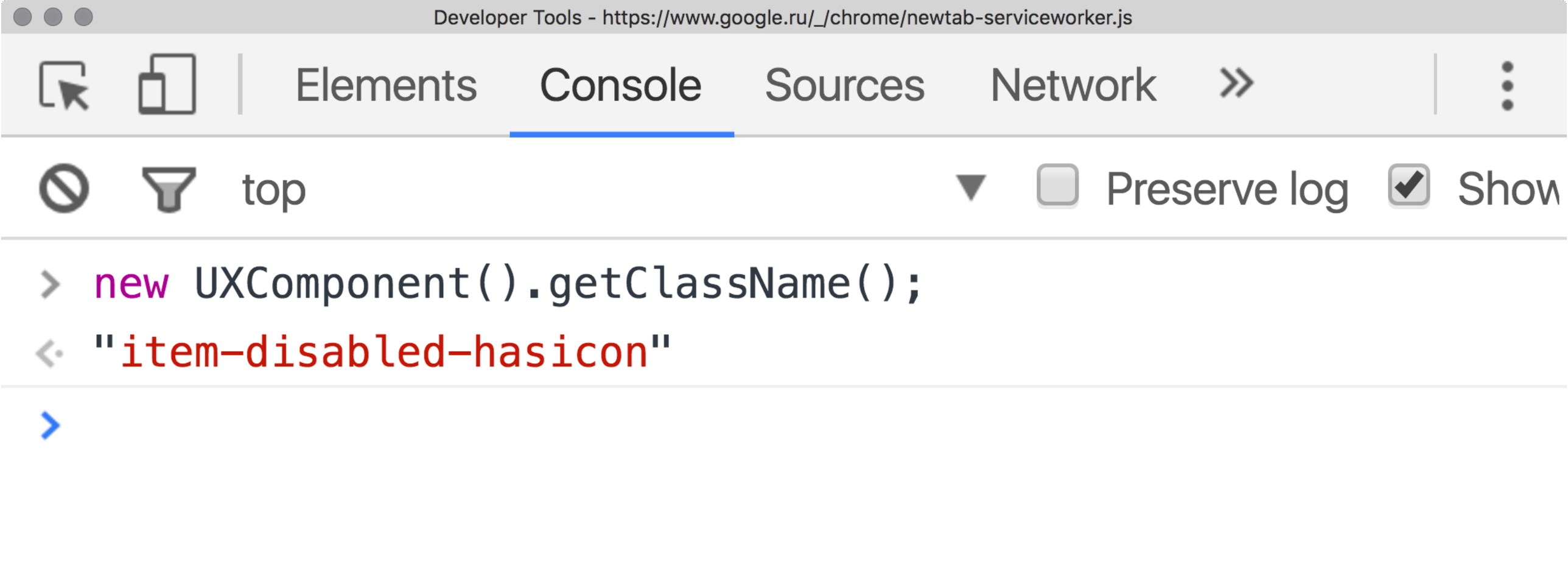
```
'use strict';

class UXComponent {
  static get State() {
    return {
      DISABLED: 0x01,
      FOCUSED: 0x02,
      HOVERED: 0x04,
      ACTIVE: 0x08,
      HAS_ICON: 0x10
    };
  }

  static get StateClassName() {
    return new Map([
      [UXComponent.State.DISABLED, 'item-disabled'],
      [UXComponent.State.FOCUCED, 'item-focused'],
      [UXComponent.State.HOVERED, 'item-hovered'],
      [UXComponent.State.ACTIVE, 'item-active'],
      [UXComponent.State.HAS_ICON, 'item-hasicon']
    ]);
  }

  constructor() {
    this.state = UXComponent.State.DISABLED | UXComponent.State.HAS_ICON;
  }

  getClassName() {
    return UXComponent.StateClassName.get(this.state);
  }
}

```

component.js — /Users/igoralexeenko/dev/ha/demo.js

component.js

```
'use strict';

class UXComponent {
  static get State() {
    return {
      DISABLED: 0x01,
      FOCUSED: 0x02,
      HOVERED: 0x04,
      ACTIVE: 0x08,
      HAS_ICON: 0x10
    };
  }

  static get StateClassName() {
    return new Map([
      [UXComponent.State.DISABLED, 'item-disabled'],
      [UXComponent.State.FOCUCED, 'item-focused'],
      [UXComponent.State.HOVERED, 'item-hovered'],
      [UXComponent.State.ACTIVE, 'item-active'],
      [UXComponent.State.HAS_ICON, 'item-hasicon'],
      [UXComponent.State.DISABLED | UXComponent.State.HAS_ICON, 'item-disabled-hasicon']
    ]);
  }

  constructor() {
    this.state = UXComponent.State.DISABLED | UXComponent.State.HAS_ICON;
  }

  getClassName() {
    return UXComponent.StateClassName.get(this.state);
  }
}
```

Line 0 | File 0 | Project 0 | No Issues | component.js | 21:80 | LF | UTF-8 | JavaScript | master | +33


Developer Tools - https://www.google.ru/_/chrome/newtab-serviceworker.js
Elements
Console
Sources
Network
top
Preserve log
Show
new UXComponent().getClassName();
"item-disabled-hasicon"

Полученный код содержит всего пять (одну) управляющих строк. Остальной код — статические маппинги. Вне зависимости от размеров маппинга (количества состояний) размер управляющего кода меняться не будет

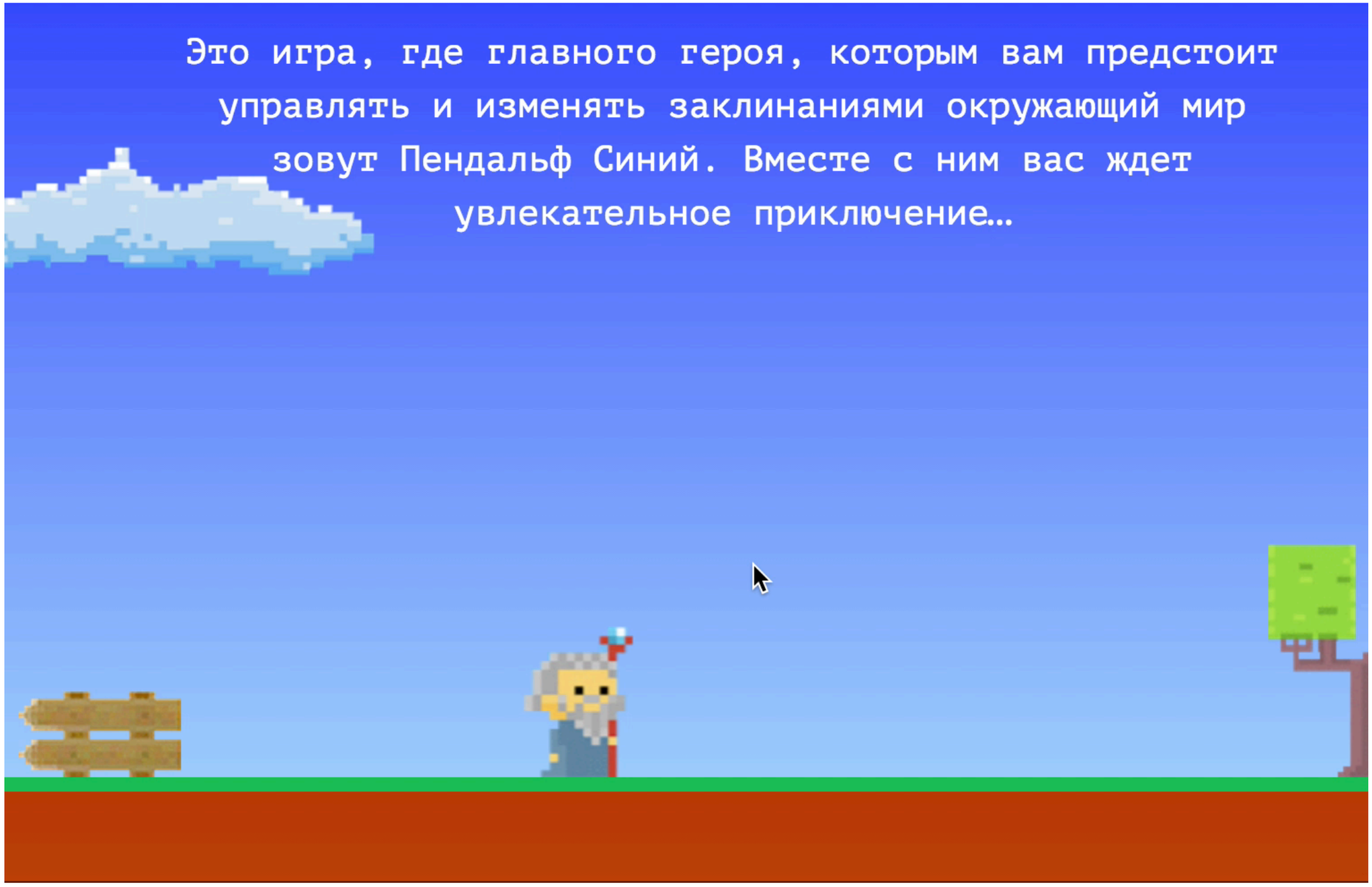
Декларативный стиль

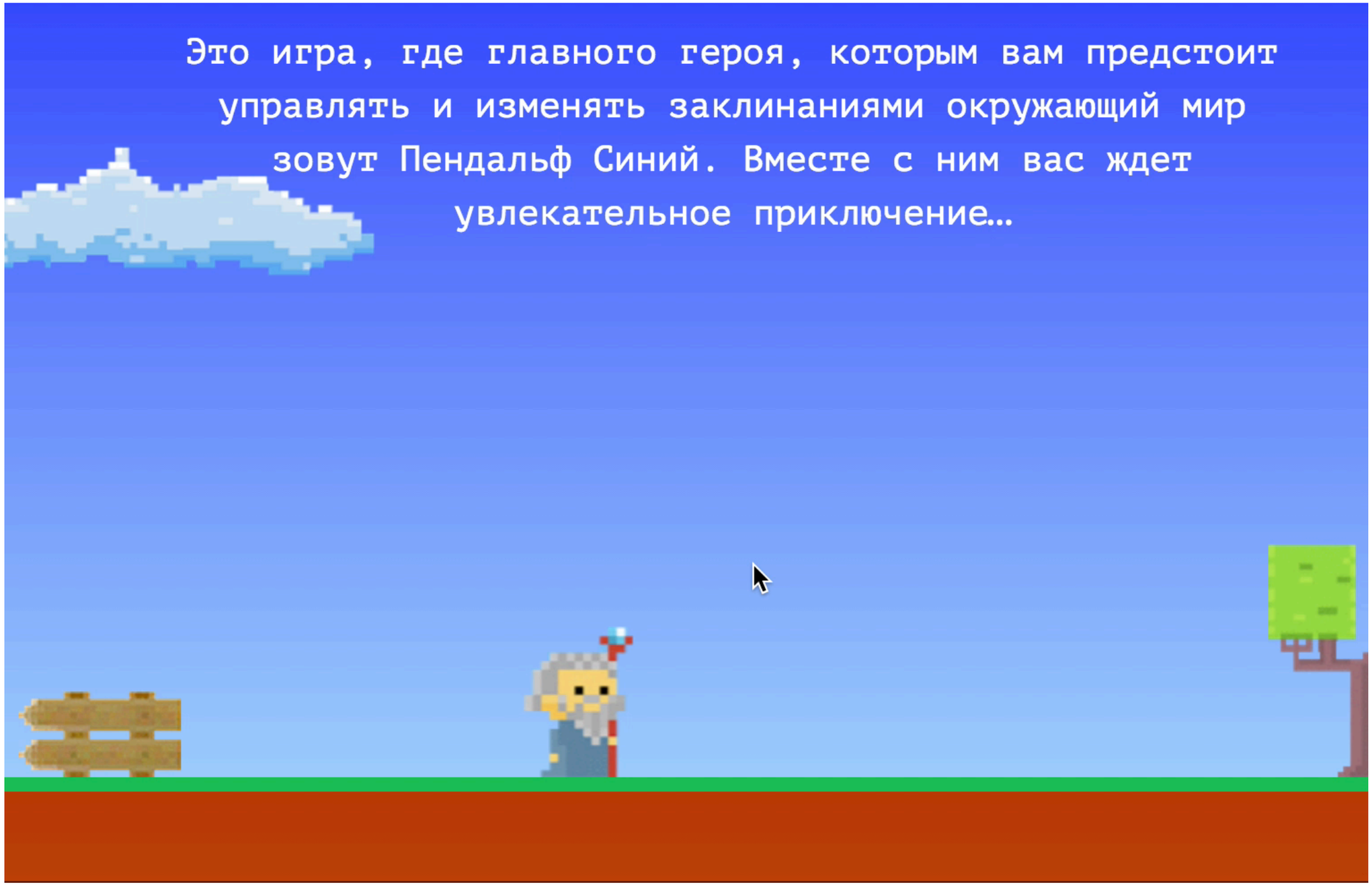
мы описываем не последовательность шагов (императивный стиль), а реакцию на изменение состояния

Декларативный стиль

мы описываем н агов (императивный стиль), а реакцию на изменение состояния

А что если в мапах использовать не статические значения, а функции?


Это игра, где главного героя, которым вам предстоит
управлять и изменять заклинаниями окружающий мир
зовут Пендальф Синий. Вместе с ним вас ждет
увлекательное приключение...


Это игра, где главного героя, которым вам предстоит
управлять и изменять заклинаниями окружающий мир
зовут Пендальф Синий. Вместе с ним вас ждет
увлекательное приключение...

main.js — /Users/igoralexeenko/dev/sandbox

main.js

```
'use strict';

class GameCharacter {
  static get Direction() {
    return {
      NULL: 0x00,
      TOP: 0x01,
      RIGHT: 0x02,
      LEFT: 0x04
    };
  }
}

```

Line 0 File 0 Project 0 ✔ No Issues main.js 9:17 LF UTF-8 JavaScript

main.js — /Users/igoralexeenko/dev/sandbox

main.js

```
'use strict';

class GameCharacter {
  static get Direction() {
    return {
      NULL: 0x00,
      TOP: 0x01,
      RIGHT: 0x02,
      LEFT: 0x04
    };
  }

  constructor() {
    this.character = {
      x: 0,
      y: 0,
      direction: GameCharacter.Direction.NULL,
      verticalSpeed: 1,
      horizontalSpeed: 1
    };
  }
}

```

Line 0 File 0 Project 0 ✔ No Issues main.js 20:7 LF UTF-8 JavaScript

main.js

```
'use strict';

class GameCharacter {
  static get Direction() {
    return {
      NULL: 0x00,
      TOP: 0x01,
      RIGHT: 0x02,
      LEFT: 0x04
    };
  }

  constructor() {
    this.character = {
      x: 0,
      y: 0,
      direction: GameCharacter.Direction.NULL,
      verticalSpeed: 1,
      horizontalSpeed: 1
    };
  }

  setDirection(direction) {
    this.character.direction = this.character.direction | direction;

    switch (direction) {
      case Direction.TOP:
        this.character.direction = this.character.direction & ~Direction.BOTTOM;
        break;

      case Direction.RIGHT:
        this.character.direction = this.character.direction & ~Direction.LEFT;
        break;
```

main.js

```
'use strict';

class GameCharacter {
  static get Direction() {
    return {
      NULL: 0x00,
      TOP: 0x01,
      RIGHT: 0x02,
      LEFT: 0x04
    };
  }

  static get DirectionBehaviour() {

  }

  constructor() {
    this.character = {
      x: 0,
      y: 0,
      direction: GameCharacter.Direction.NULL,
      verticalSpeed: 1,
      horizontalSpeed: 1
    };
  }

  setDirection(direction) {
    this.character.direction = this.character.direction | direction;

    switch (direction) {
      case Direction.TOP:
        this.character.direction = this.character.direction & ~Direction.BOTTOM;
        break;
```

main.js

```
      LEFT: 0x04
    };
  }

  static get DirectionBehaviour() {
    return new Map([

    ]);
  }

  constructor() {
    this.character = {
      x: 0,
      y: 0,
      direction: GameCharacter.Direction.NULL,
      verticalSpeed: 1,
      horizontalSpeed: 1
    };
  }

  setDirection(direction) {
    this.character.direction = this.character.direction | direction;

    switch (direction) {
      case Direction.TOP:
        this.character.direction = this.character.direction & ~Direction.BOTTOM;
        break;

      case Direction.RIGHT:
        this.character.direction = this.character.direction & ~Direction.LEFT;
        break;

      case Direction.BOTTOM:
```

main.js

```
      LEFT: 0x04
    };
  }

  static get DirectionBehaviour() {
    return new Map([
      [GameCharacter.Direction.TOP, []
    ]);
  }

  constructor() {
    this.character = {
      x: 0,
      y: 0,
      direction: GameCharacter.Direction.NULL,
      verticalSpeed: 1,
      horizontalSpeed: 1
    };
  }

  setDirection(direction) {
    this.character.direction = this.character.direction | direction;

    switch (direction) {
      case Direction.TOP:
        this.character.direction = this.character.direction & ~Direction.BOTTOM;
        break;

      case Direction.RIGHT:
        this.character.direction = this.character.direction & ~Direction.LEFT;
        break;

      case Direction.BOTTOM:
```

main.js

```
      LEFT: 0x04
    };
  }

  static get DirectionBehaviour() {
    return new Map([
      [GameCharacter.Direction.TOP, (character) => {
        return Object.assign({}, character, {
          y: character.y + character.verticalSpeed
        });
      }]
    ]);
  }

  constructor() {
    this.character = {
      x: 0,
      y: 0,
      direction: GameCharacter.Direction.NULL,
      verticalSpeed: 1,
      horizontalSpeed: 1
    };
  }

  setDirection(direction) {
    this.character.direction = this.character.direction | direction;

    switch (direction) {
      case Direction.TOP:
        this.character.direction = this.character.direction & ~Direction.BOTTOM;
        break;

      case Direction.RIGHT:
```

main.js

```
      LEFT: 0x04
    };
  }

  static get DirectionBehaviour() {
    return new Map([
      [GameCharacter.Direction.TOP, (character) => {
        return Object.assign({}, character, {
          y: character.y + character.verticalSpeed
        });
      }],

      [GameCharacter.Direction.RIGHT, (character) => {
        return Object.assign({}, character, {
          x: character.x + character.horizontalSpeed
        });
      ],

      [GameCharacter.Direction.BOTTOM, (character) => {
        return Object.assign({}, character, {
          y: character.y - character.verticalSpeed
        });
      ],

      [GameCharacter.Direction.LEFT, (character) => {
        return Object.assign({}, character, {
          x: character.x - character.horizontalSpeed
        });
      ]
    ]);
  }

  constructor() {
```

main.js

```
    this.character = {
      x: 0,
      y: 0,
      direction: GameCharacter.Direction.NULL,
      verticalSpeed: 1,
      horizontalSpeed: 1
    };
  }

  update() {
    GameCharacter.DirectionBehaviour.forEach((action, direction) => {
      if (!!(direction & this.character.direction)) {
        this.character = action(this.character);
      }
    });
  }

  setDirection(direction) {
    this.character.direction = this.character.direction | direction;

    switch (direction) {
      case Direction.TOP:
        this.character.direction = this.character.direction & ~Direction.BOTTOM;
        break;

      case Direction.RIGHT:
        this.character.direction = this.character.direction & ~Direction.LEFT;
        break;

      case Direction.BOTTOM:
        this.character.direction = this.character.direction & ~Direction.TOP;
        break;

```

Даже Redux!

main.js — /Users/igoralexeenko/dev/sandbox

main.js

```
1   const ActionType = {
2     SOMETHING: 0x00,
3     ANOTHER: 0x02,
4     THIRD: 0x04
5   };
6
7   let reducer = (state, action) => {
8     switch (action.type) {
9       case ActionType.SOMETHING: return doSomething(state, action);
10      case ActionType.ANOTHER: return doAnother(state, action);
11      case ActionType.THIRD: return doSomething(state, action);
12    }
13  }
14
```

Line 0 | File 0 | Project 0 | No Issues | main.js | 11:62 | LF | UTF-8 | JavaScript

main.js — /Users/igoralexeenko/dev/sandbox

main.js

```
1   const ActionType = {
2     SOMETHING: 0x00,
3     ANOTHER: 0x02,
4     THIRD: 0x04
5   };
6
7   let reducer = (state, action) => {
8
9   }
10
```

Line 0 File 0 Project 0 ✔ No Issues main.js 8:1 LF UTF-8 JavaScript

main.js — /Users/igoralexeenko/dev/sandbox

main.js

```
1   const ActionType = {
2     SOMETHING: 0x00,
3     ANOTHER: 0x02,
4     THIRD: 0x04
5   };
6
7   const ActionMap = new Map([
8     [ActionType.SOMETHING, (state, action) => {
9       return doSomething(state, action);
10    }]
11  ]);
12
13  let reducer = (state, action) => {
14
15  }
16
```

Line 0 File 0 Project 0 ✔ No Issues main.js 9:39 LF UTF-8 JavaScript

main.js

```
const ActionType = {
  SOMETHING: 0x00,
  ANOTHER: 0x02,
  THIRD: 0x04
};

const ActionMap = new Map([
  [ActionType.SOMETHING, (state, action) => {
    return doSomething(state, action);
  }],
  [ActionType.ANOTHER, (state, action) => {
    return doAnother(state, action);
  }],
  [ActionType.THIRD, (state, action) => {
    return doThird(state, action);
  }]
]);

let reducer = (state, action) => {

}

```

main.js

```
const ActionType = {
  SOMETHING: 0x00,
  ANOTHER: 0x02,
  THIRD: 0x04
};

const ActionMap = new Map([
  [ActionType.SOMETHING, (state, action) => {
    return doSomething(state, action);
  }],
  [ActionType.ANOTHER, (state, action) => {
    return doAnother(state, action);
  }],
  [ActionType.THIRD, (state, action) => {
    return doThird(state, action);
  }]
]);

let reducer = (state, action) => {
  ActionMap.forEach((actionReducer, actionType) => {
    if ((action.type & actionType)) {
      state = actionReducer(state, action);
    }
  });

  return state;
}
```

main.js

```
const ActionType = {
  SOMETHING: 0x00,
  ANOTHER: 0x02,
  THIRD: 0x04
};

const ActionMap = new Map([
  [ActionType.SOMETHING, (state, action) => {
    return doSomething(state, action);
  }],
  [ActionType.ANOTHER, (state, action) => {
    return doAnother(state, action);
  }],
  [ActionType.THIRD, (state, action) => {
    return doThird(state, action);
  }]
]);

let reducer = (state, action) => {
  ActionMap.forEach((actionReducer, actionType) => {
    if ((action.type & actionType)) {
      state = actionReducer(state, action);
    }
  });

  return state;
}

let store = createStore(reducer, {});

```

main.js

```
const ActionType = {
  SOMETHING: 0x00,
  ANOTHER: 0x02,
  THIRD: 0x04
};

const ActionMap = new Map([
  [ActionType.SOMETHING, (state, action) => {
    return doSomething(state, action);
  }],
  [ActionType.ANOTHER, (state, action) => {
    return doAnother(state, action);
  }],
  [ActionType.THIRD, (state, action) => {
    return doThird(state, action);
  }]
]);

let reducer = (state, action) => {
  ActionMap.forEach((actionReducer, actionType) => {
    if ((action.type & actionType)) {
      state = actionReducer(state, action);
    }
  });

  return state;
}

let store = createStore(reducer, {});

store.dispatch({ type: ActionType.SOMETHING });
```

main.js

```
const ActionType = {
  SOMETHING: 0x00,
  ANOTHER: 0x02,
  THIRD: 0x04
};

const ActionMap = new Map([
  [ActionType.SOMETHING, (state, action) => {
    return doSomething(state, action);
  }],
  [ActionType.ANOTHER, (state, action) => {
    return doAnother(state, action);
  }],
  [ActionType.THIRD, (state, action) => {
    return doThird(state, action);
  }]
]);

let reducer = (state, action) => {
  ActionMap.forEach((actionReducer, actionType) => {
    if ((action.type & actionType)) {
      state = actionReducer(state, action);
    }
  });

  return state;
}

let store = createStore(reducer, {});

store.dispatch({ type: ActionType.SOMETHING | ActionType.ANOTHER });

```

И кроме этого есть еще

И кроме этого есть еще

- **исключающие или *(XOR)***
  используется для переключения состояния *(toggle)*

И кроме этого есть еще

- **исключающие или *(XOR)***
  используется для переключения состояния *(toggle)*

- **побитовый сдвиг влево *(<<)***

И кроме этого есть еще

- **исключающие или *(XOR)***
  используется для переключения состояния *(toggle)*

- **побитовый сдвиг влево *(<<)***

- **побитовый сдвиг вправо *(>>)***

И кроме этого есть еще

- **исключающие или *(XOR)***
  используется для переключения состояния *(toggle)*
- **побитовый сдвиг влево *(<<)***
- **побитовый сдвиг вправо *(>>)***
- **сдвиг вправо с заполнением нулями *(>>>)***

И кроме этого есть еще

- **исключающие или *(XOR)***
  используется для переключения состояния *(toggle)*

- **побитовый сдвиг влево *(<<)***

- **побитовый сдвиг вправо *(>>)***

- **сдвиг вправо с заполнением нулями *(>>>)***

- **битовые маски**
  когда одно состояние хранится в нескольких соседних битах используются маски, которые показывают единицами, в какие именно биты состояние записано *(IP-адрес и маска подсети)*

Деревья

они же связные графы без циклов

Demo

Игорь

localhost:8080

HTML | CSS | LESS | JavaScript | +

×1 | ×0.5 | ×0.75 | ×1.5 | 400px | 800px | 1000px | 1500px | 1

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4    <title></title>
5  </head>
6  <body>
7    <div id="layout"></div>
8  </body>
9  </html>
```

Demo

Игорь

localhost:8080

HTML

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
</head>
<body>
  <div id="layout"></div>
</body>
</html>
```

CSS | LESS

```
@color: white;
@height: 100px;
@min-height: calc(@height / 2);
```

JavaScript

```
body.onload = function() {
  console.log('jopa');
}
```

×1 ×0.5 ×0.75 ×1.5 400px 800px 1000px 1500px

Demo

Игорь

localhost:8080

HTML | JavaScript | +

×1 | ×0.5 | ×0.75 | ×1.5 | 400px | 800px | 1000px | 1500px | 1

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4    <title></title>
5  </head>
6  <body>
7    <div id="layout"></div>
8  </body>
9  </html>
```

CSS | LESS | +

```
1  @color: red
2
3  body {
4    color: @color;
5  }
```

Demo

Игорь

localhost:8080

HTML

```
1  <!DOCTYPE html>
2  <html>
3    <meta>
4  </html>
```

×1 ×0.5 ×0.75 ×1.5 400px 800px 1000px 1500px 100%

Требования к лейауту

Требования к лейауту

- передается с сервера как изначальное состояние; подстраивается под каждую демку индивидуально

Требования к лейауту

- передается с сервера как изначальное состояние; подстраивается под каждую демку индивидуально

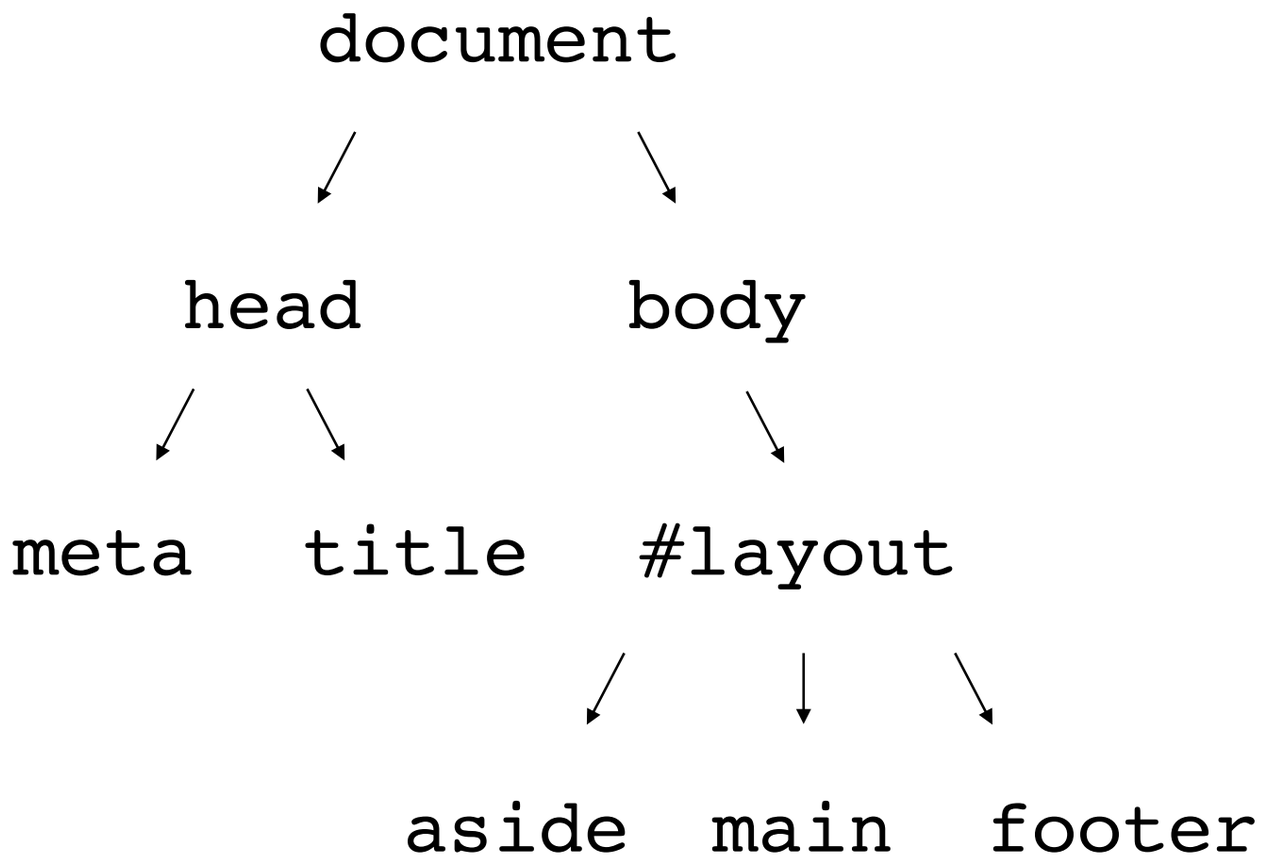
- легко настраивается: открываются новые панели, все перетаскивается, размеры изменяются

Требования к лейауту

- передается с сервера как изначальное состояние; подстраивается под каждую демку индивидуально
- легко настраивается: открываются новые панели, все перетаскивается, размеры изменяются
- сохраняется в локалсторадже и применяется заново, когда пользователь возвращается на страницу

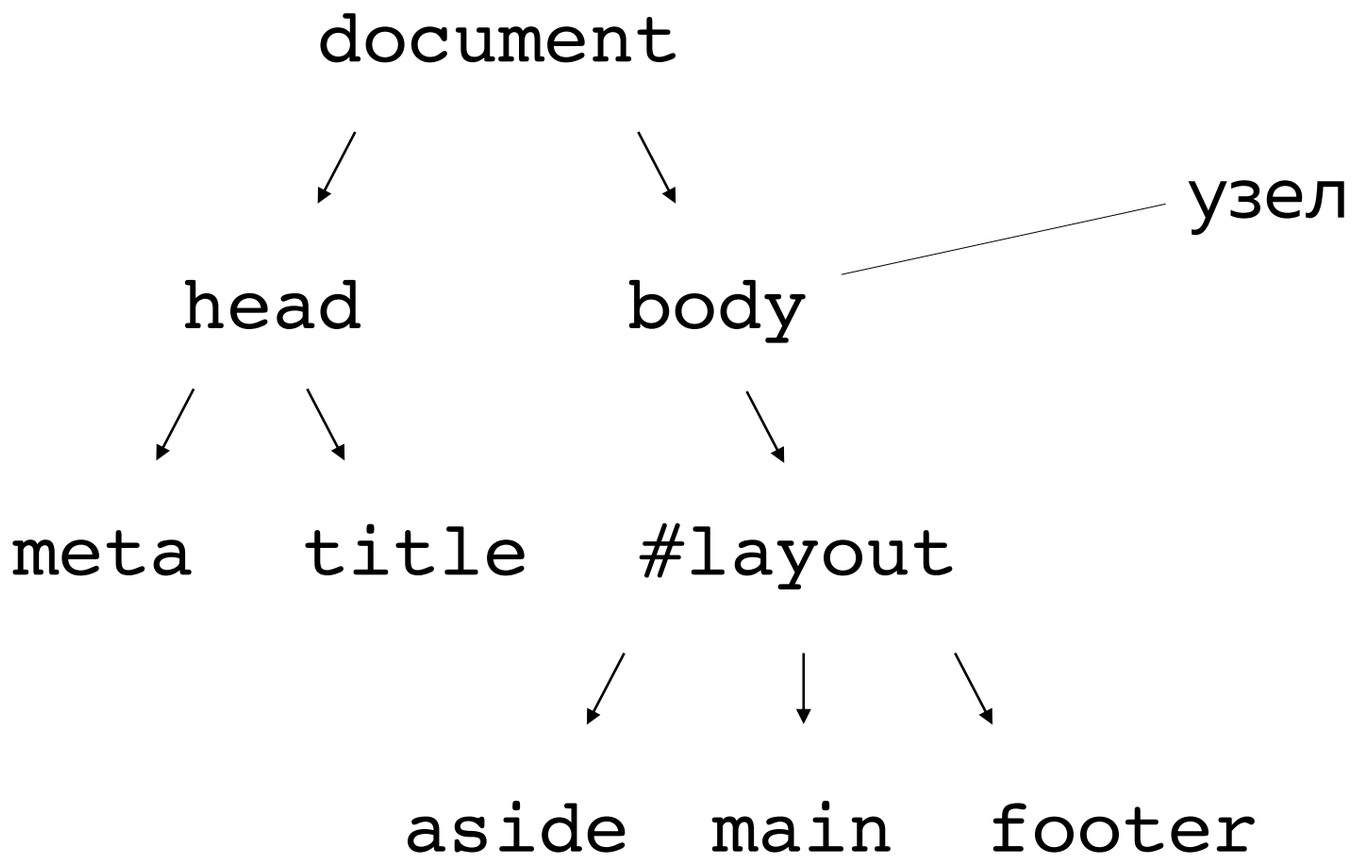
Деревья

связанная структура данных состоящая из вложенных друг в друга повторяющихся узлов. Самый наглядный пример, для фронтенд-разработчика — DOM-дерево



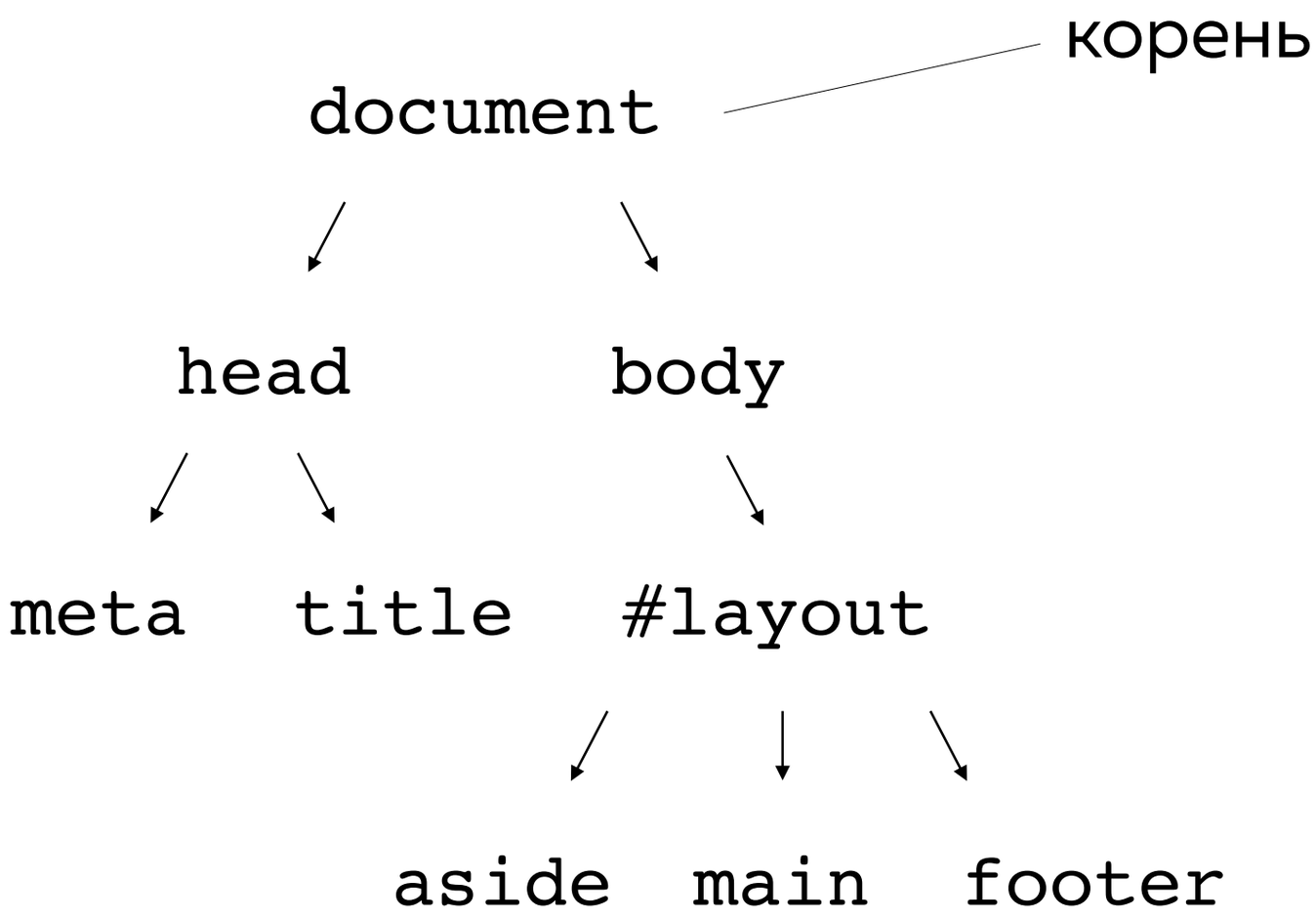
Деревья

связанная структура данных состоящая из вложенных друг в друга повторяющихся узлов. Самый наглядный пример, для фронтенд-разработчика — DOM-дерево



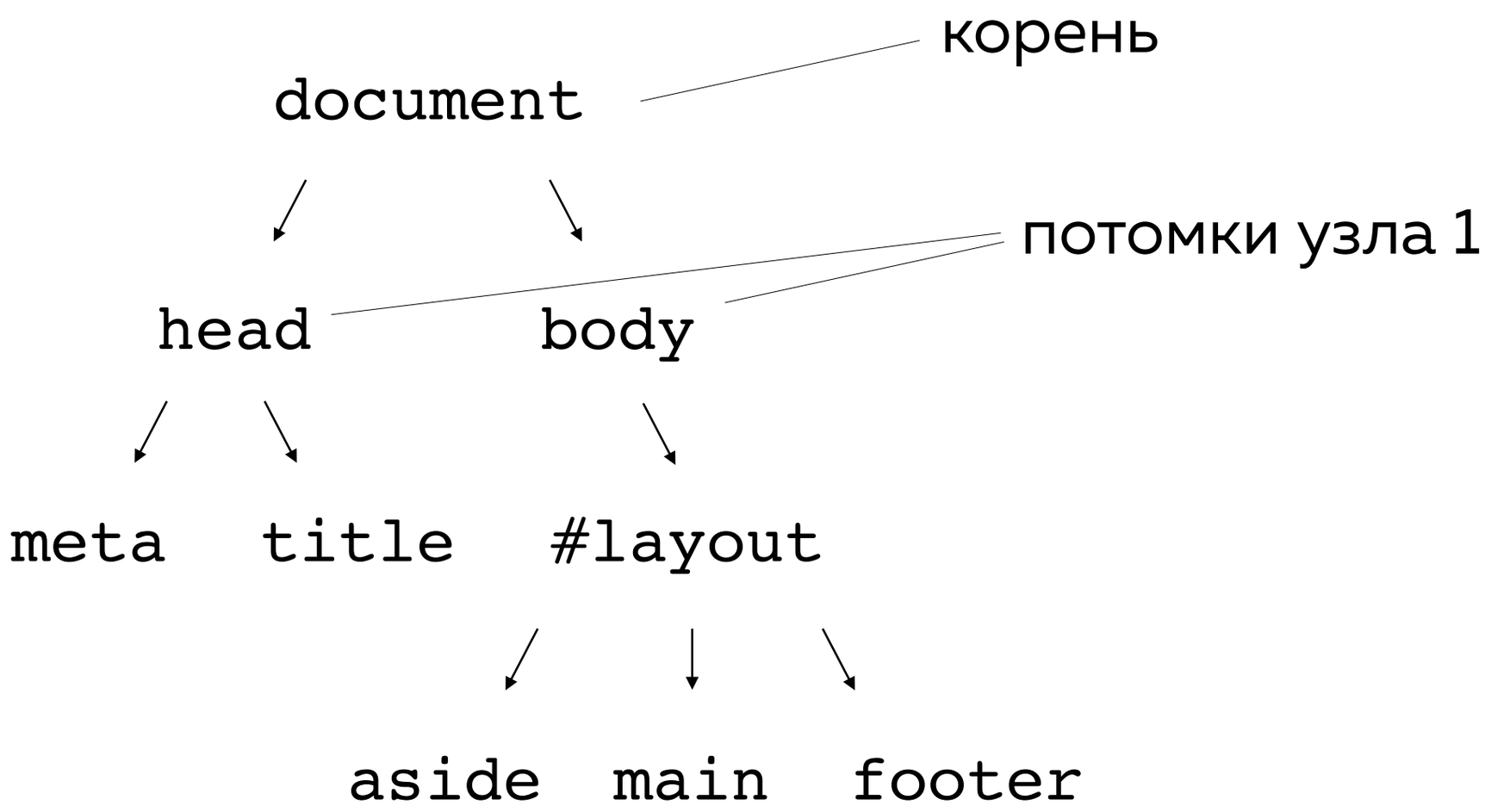
Деревья

связанная структура данных состоящая из вложенных друг в друга повторяющихся узлов. Самый наглядный пример, для фронтенд-разработчика — DOM-дерево



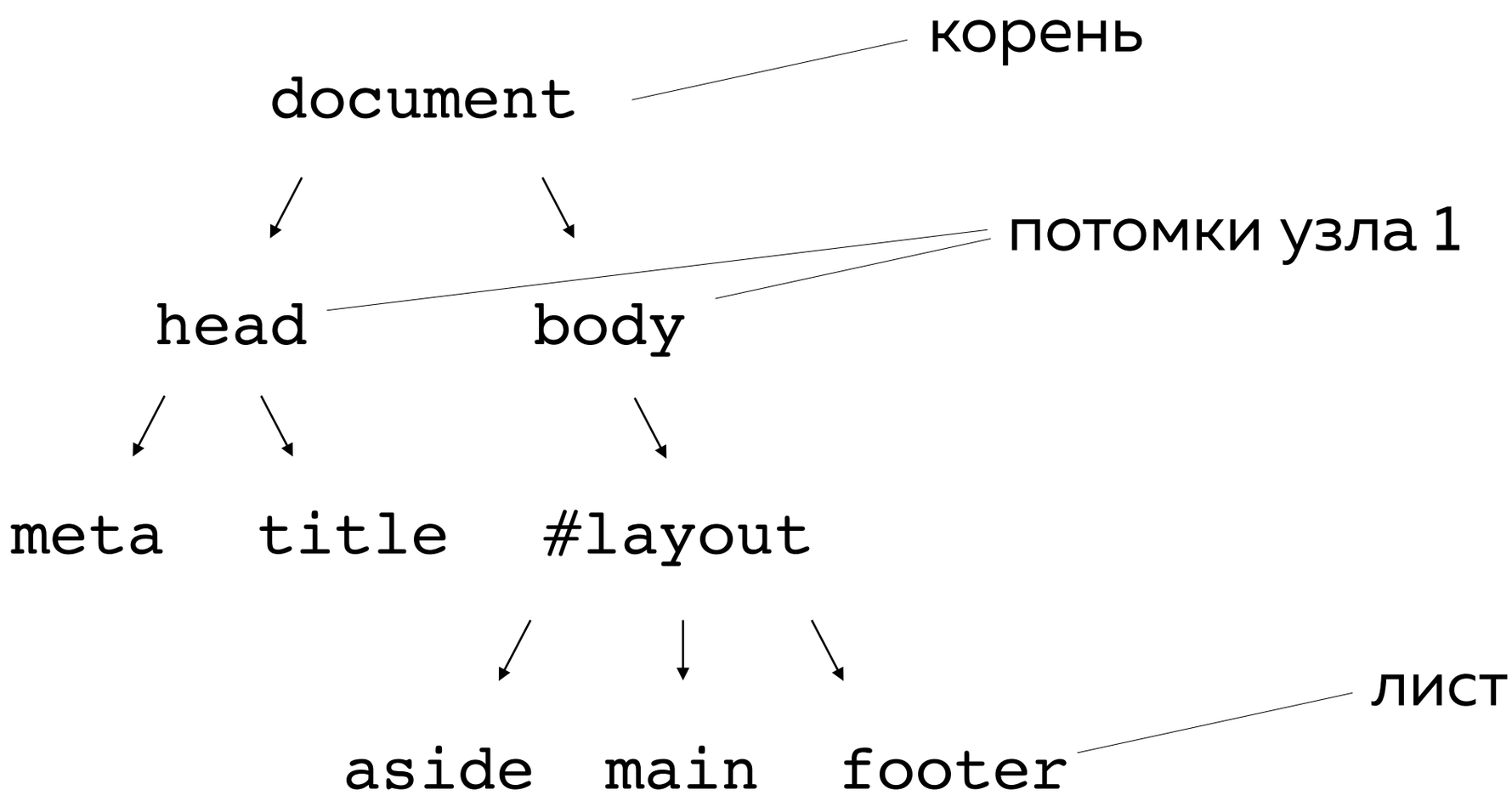
Деревья

связанная структура данных состоящая из вложенных друг в друга повторяющихся узлов. Самый наглядный пример, для фронтенд-разработчика — DOM-дерево



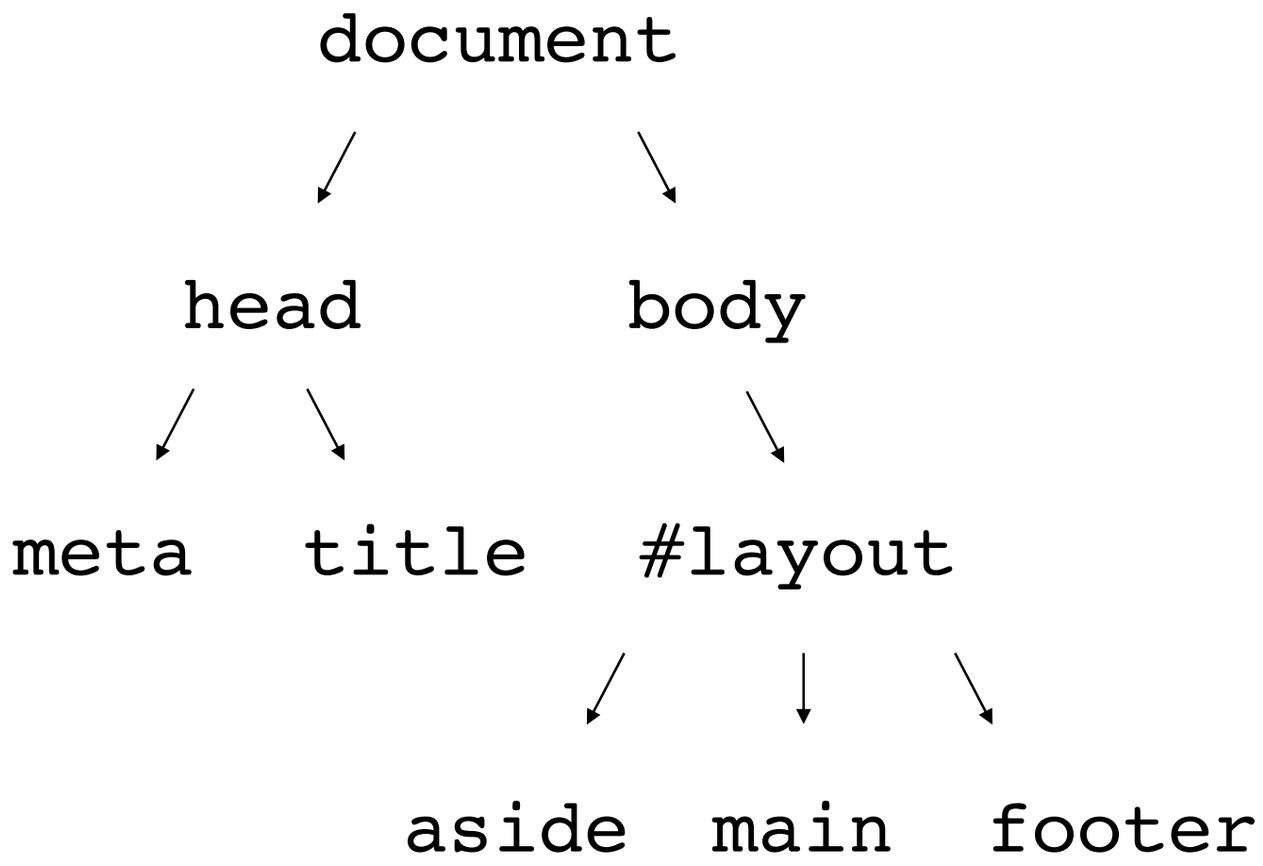
Деревья

связанная структура данных состоящая из вложенных друг в друга повторяющихся узлов. Самый наглядный пример, для фронтенд-разработчика — DOM-дерево



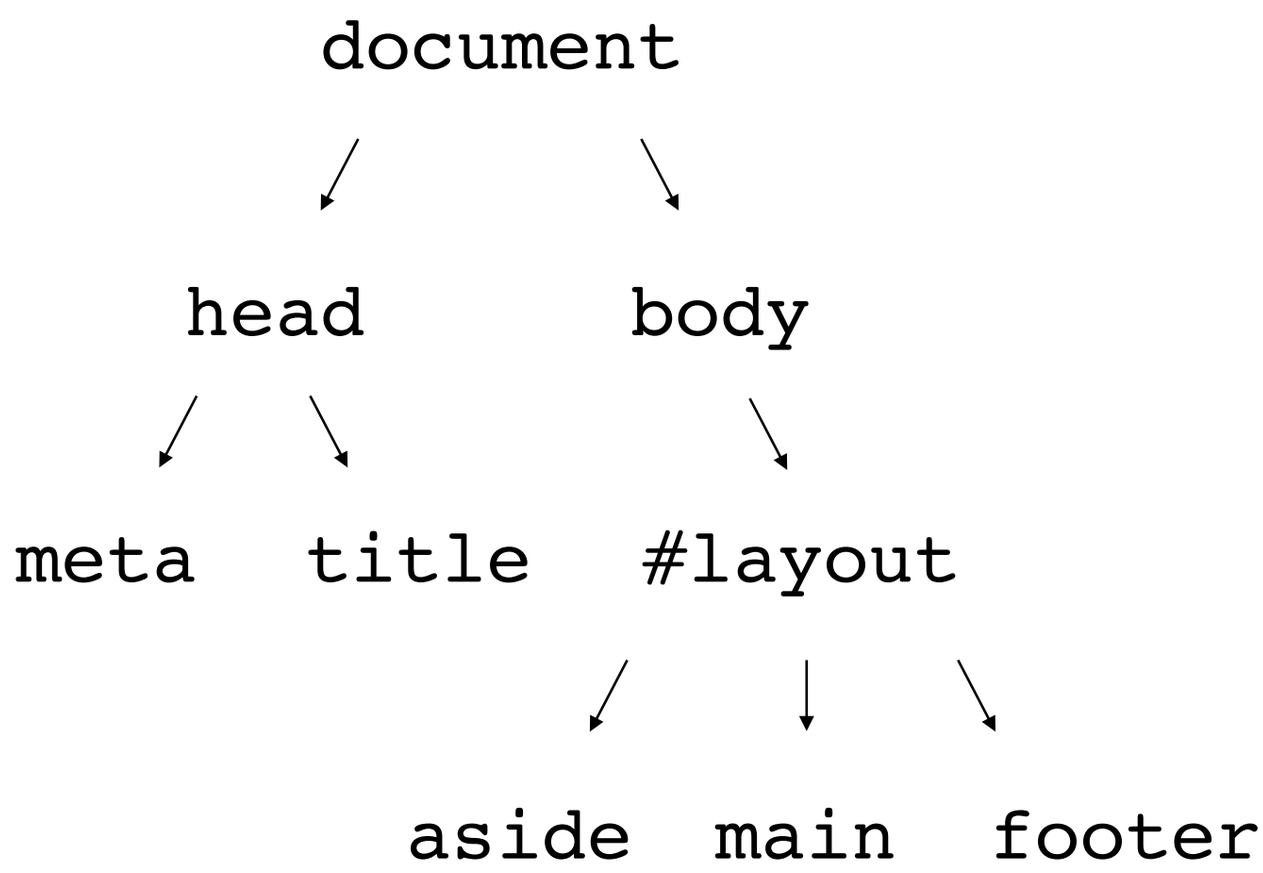
Деревья

связанная структура данных состоящая из вложенных друг в друга повторяющихся узлов. Самый наглядный пример, для фронтенд-разработчика — DOM-дерево



Деревья

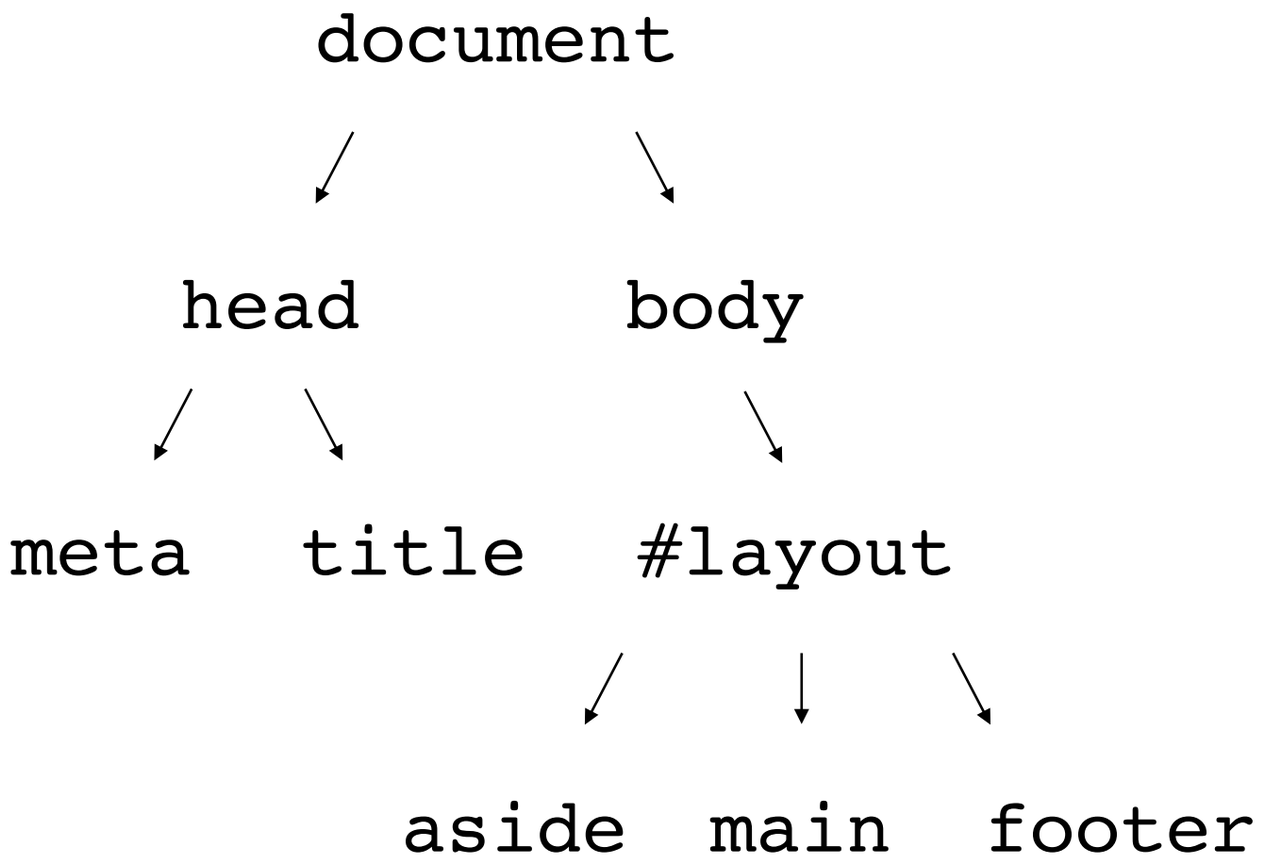
связанная структура данных состоящая из вложенных друг в друга повторяющихся узлов. Самый наглядный пример, для фронтенд-разработчика — DOM-дерево



Существует несколько способов перебора элементов дерева

Деревья

связанная структура данных состоящая из вложенных друг в друга повторяющихся узлов. Самый наглядный пример, для фронтенд-разработчика — DOM-дерево

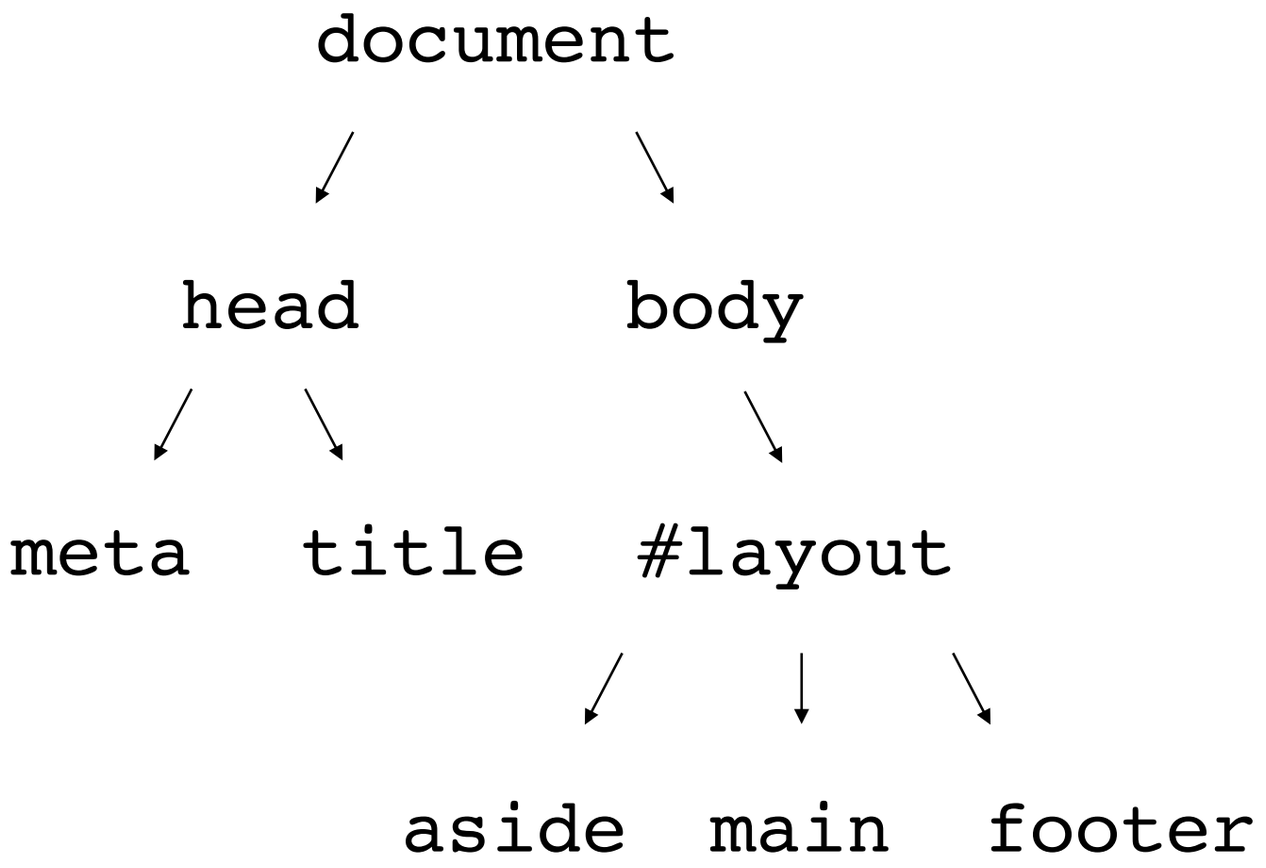


Существует несколько способов перебора элементов дерева

- DFS (поиск в глубину)

Деревья

связанная структура данных состоящая из вложенных друг в друга повторяющихся узлов. Самый наглядный пример, для фронтенд-разработчика — DOM-дерево

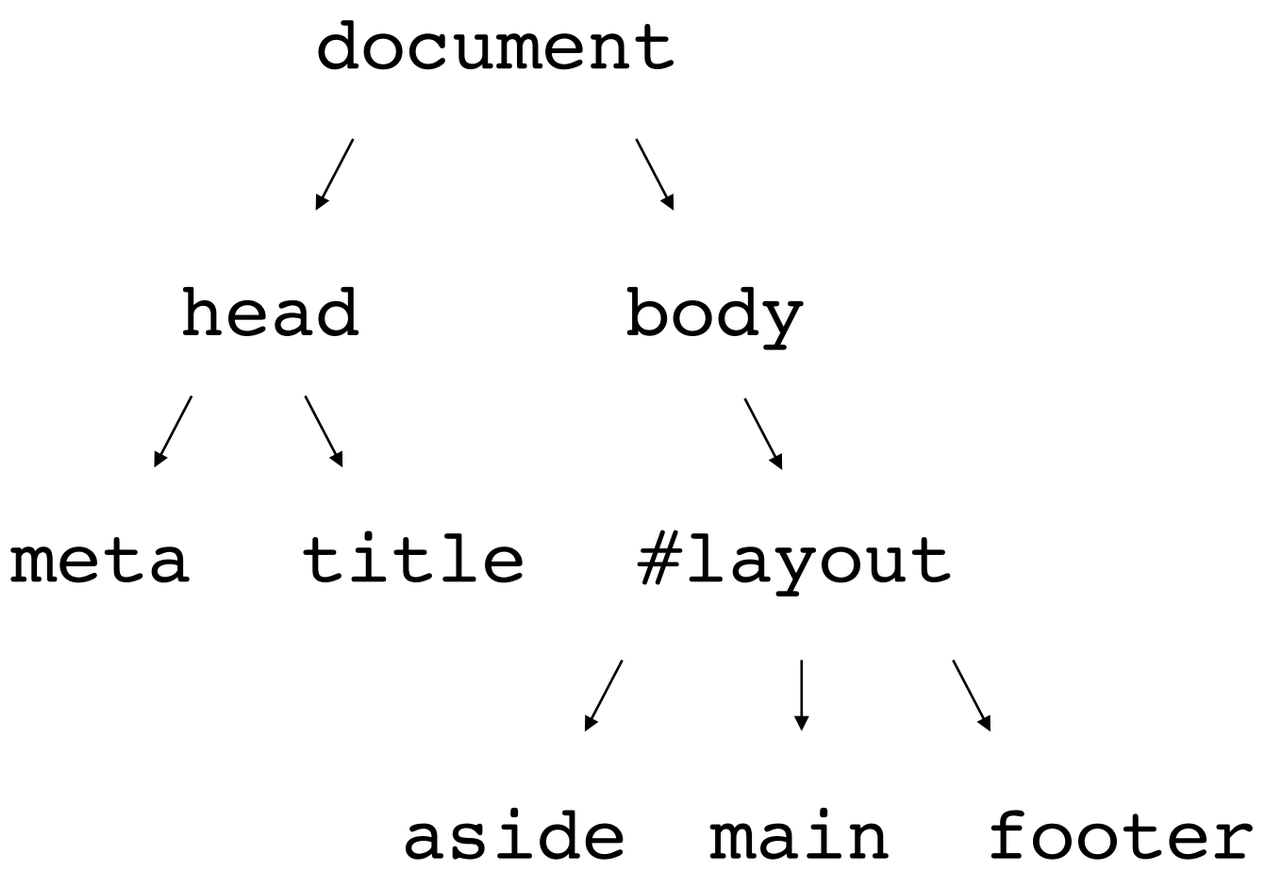


Существует несколько способов перебора элементов дерева

- DFS (поиск в глубину)
  - прямой

Деревья

связанная структура данных состоящая из вложенных друг в друга повторяющихся узлов. Самый наглядный пример, для фронтенд-разработчика — DOM-дерево

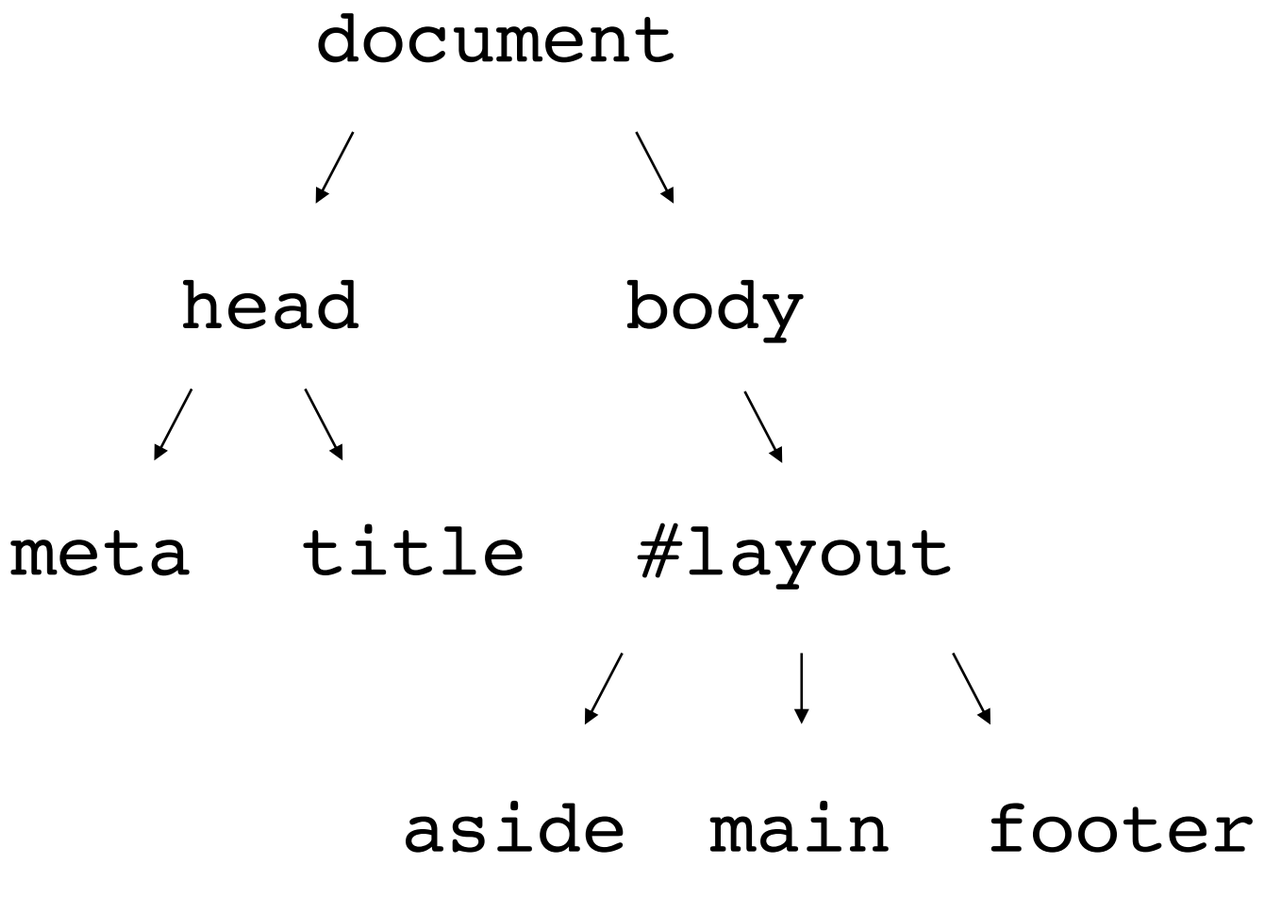


Существует несколько способов перебора элементов дерева

- DFS (поиск в глубину)
  - прямой
  - обратный

Деревья

связанная структура данных состоящая из вложенных друг в друга повторяющихся узлов. Самый наглядный пример, для фронтенд-разработчика — DOM-дерево

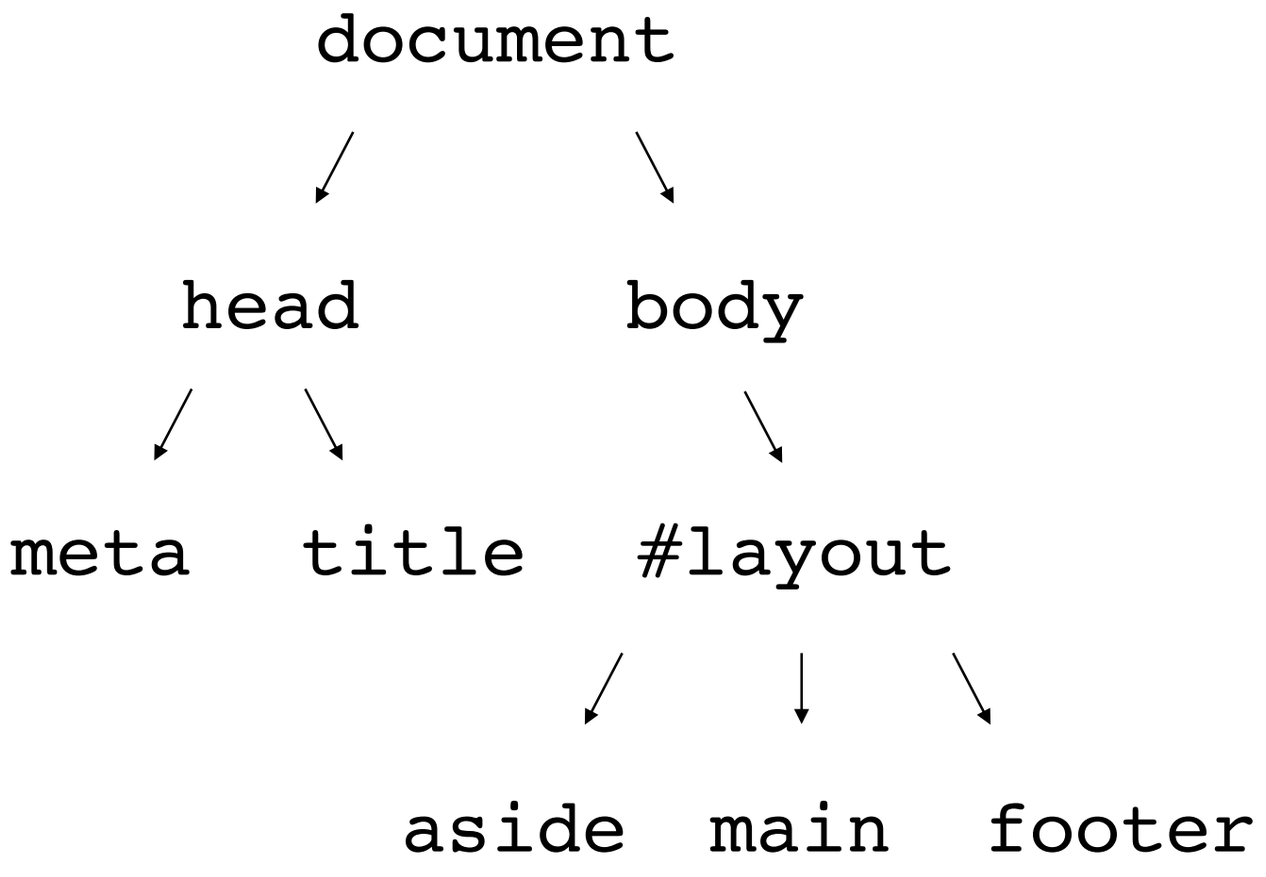


Существует несколько способов перебора элементов дерева

- DFS (поиск в глубину)
  - прямой
  - обратный
  - симметричный

Деревья

связанная структура данных состоящая из вложенных друг в друга повторяющихся узлов. Самый наглядный пример, для фронтенд-разработчика — DOM-дерево

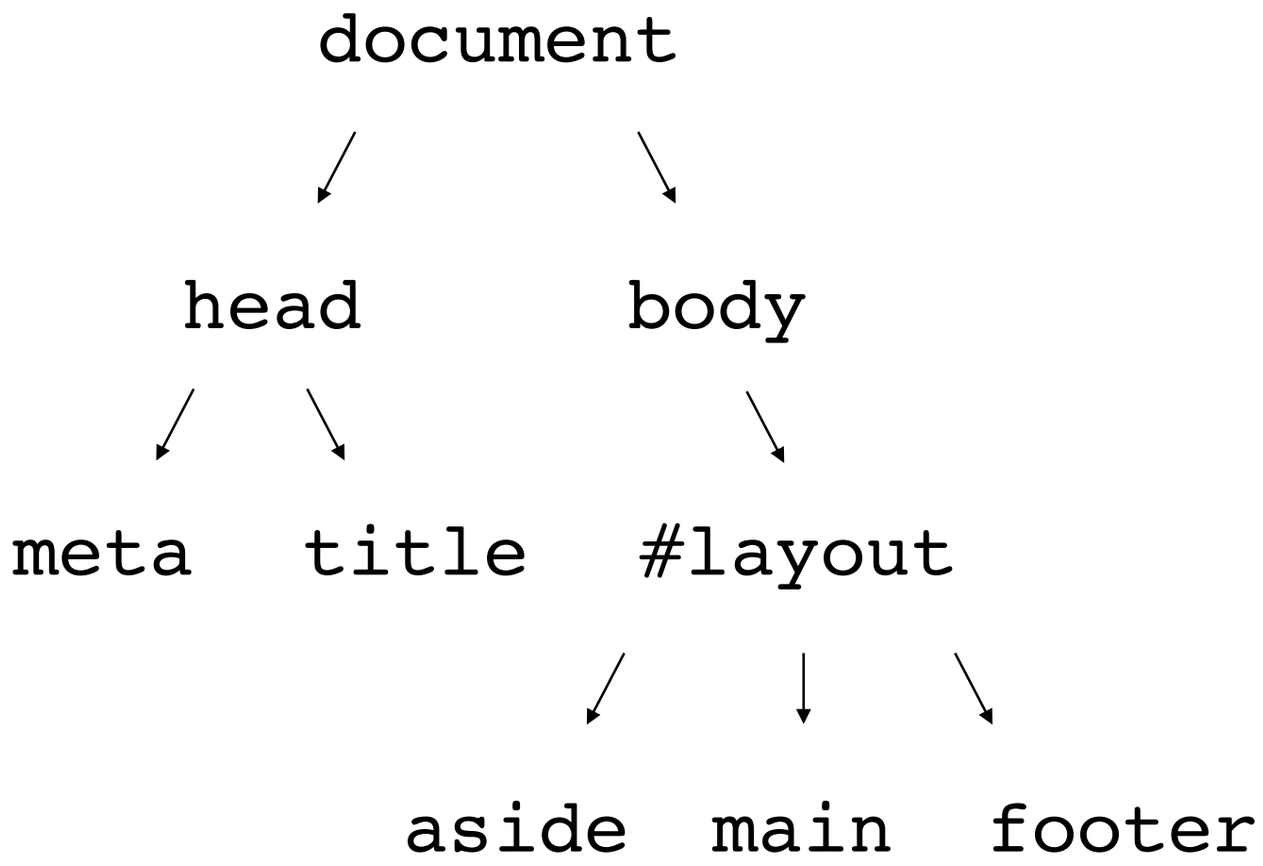


Существует несколько способов перебора элементов дерева

- DFS (поиск в глубину)
  - прямой
  - обратный
  - симметричный
- BFS (поиск в ширину)

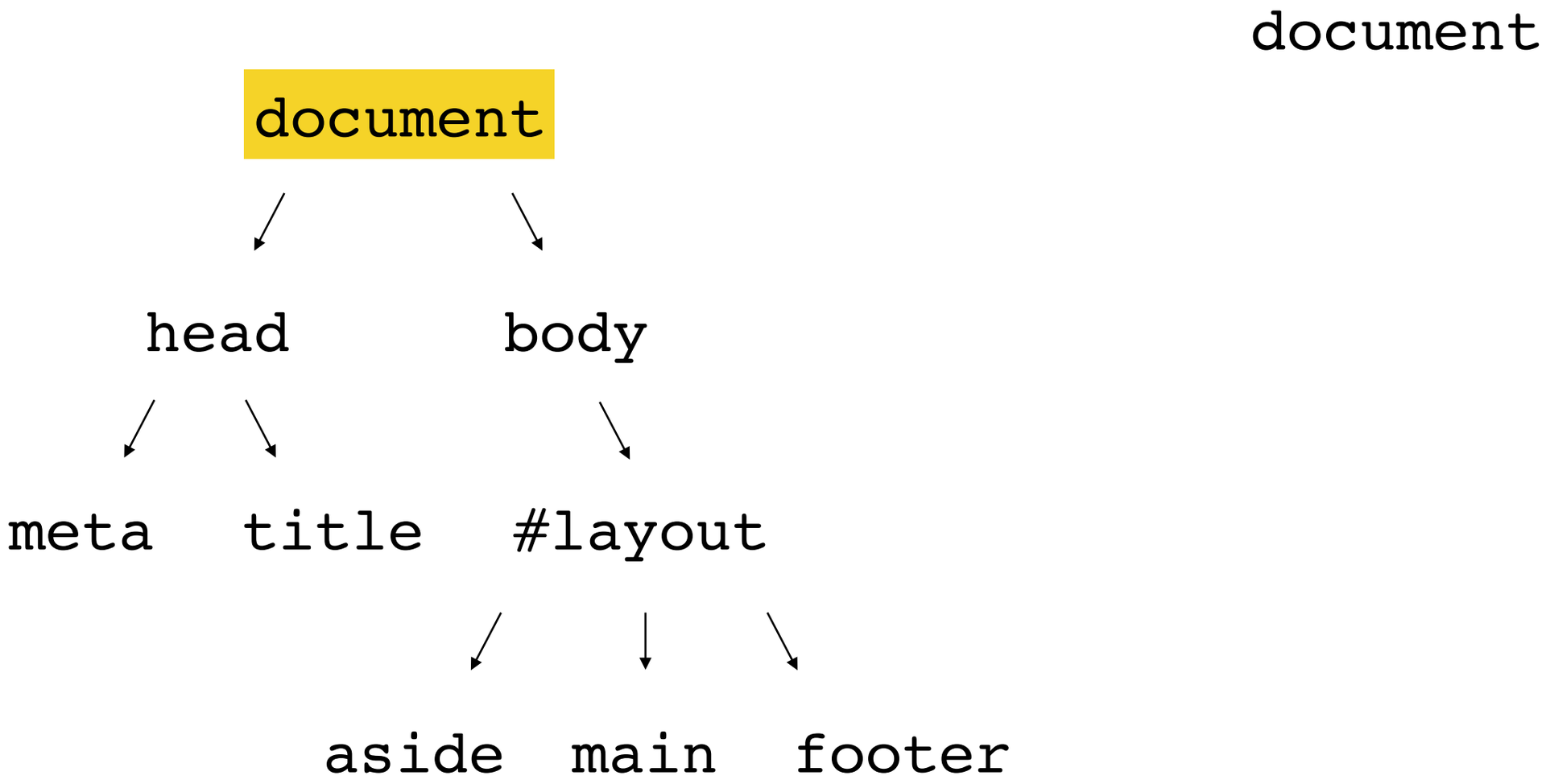
Прямой обход в глубину

Сначала посещается узел, а потом его потомки



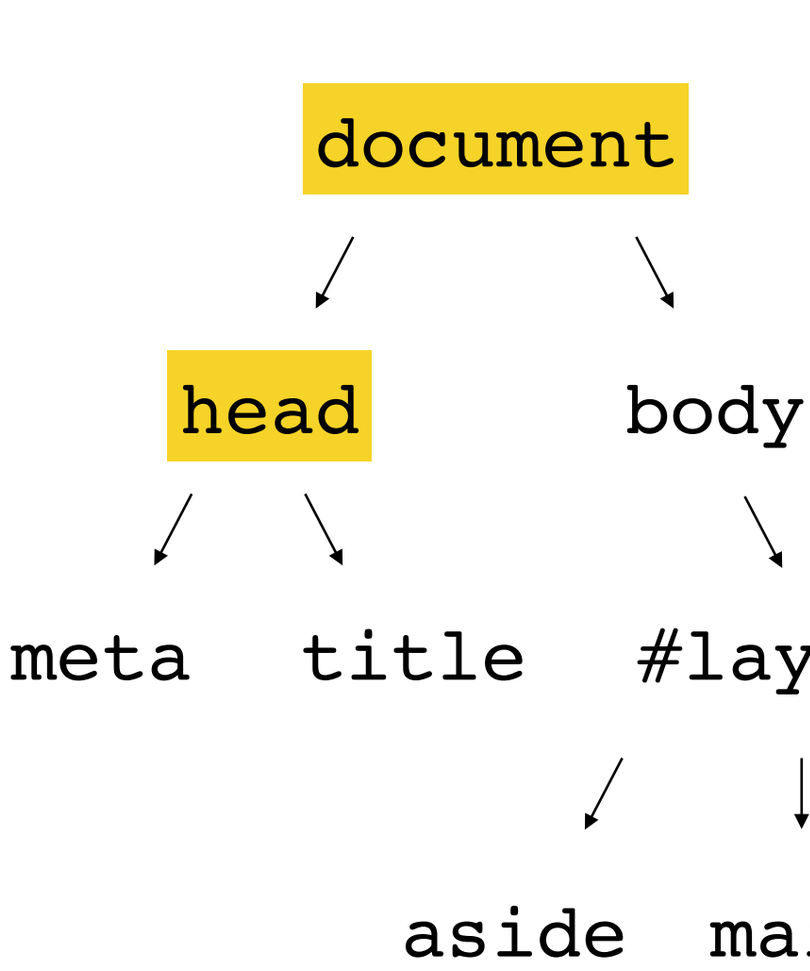
Прямой обход в глубину

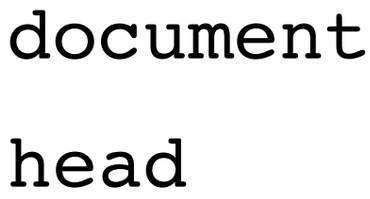
Сначала посещается узел, а потом его потомки



Прямой обход в глубину

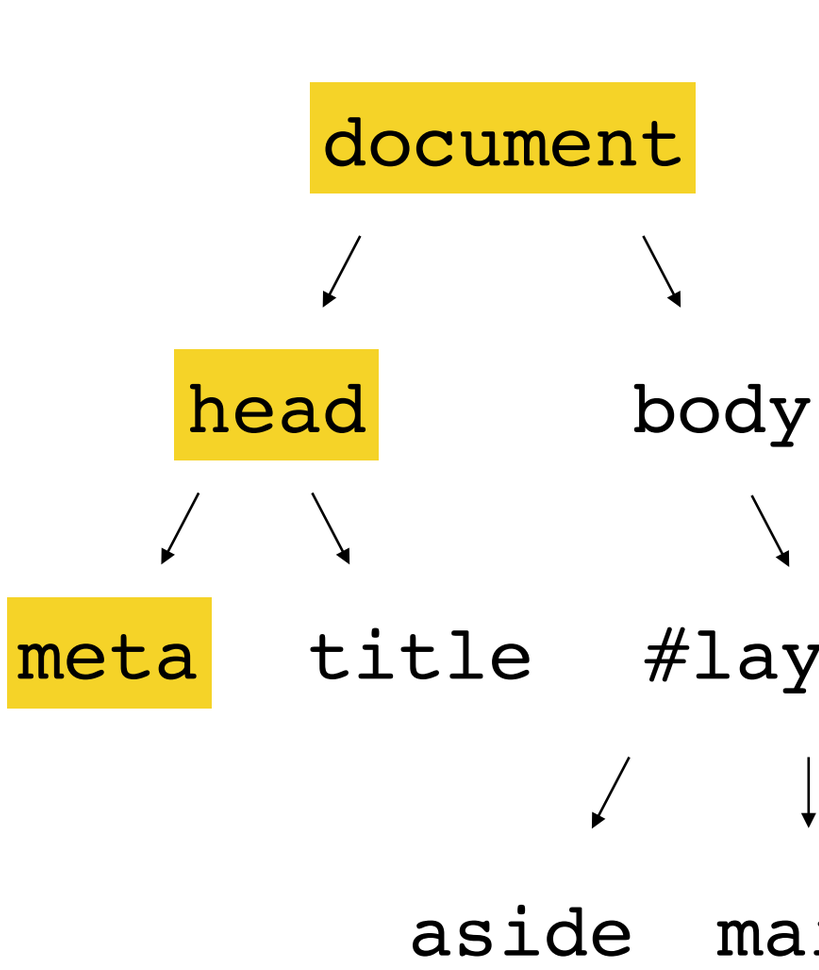
Сначала посещается узел, а потом его потомки





Прямой обход в глубину

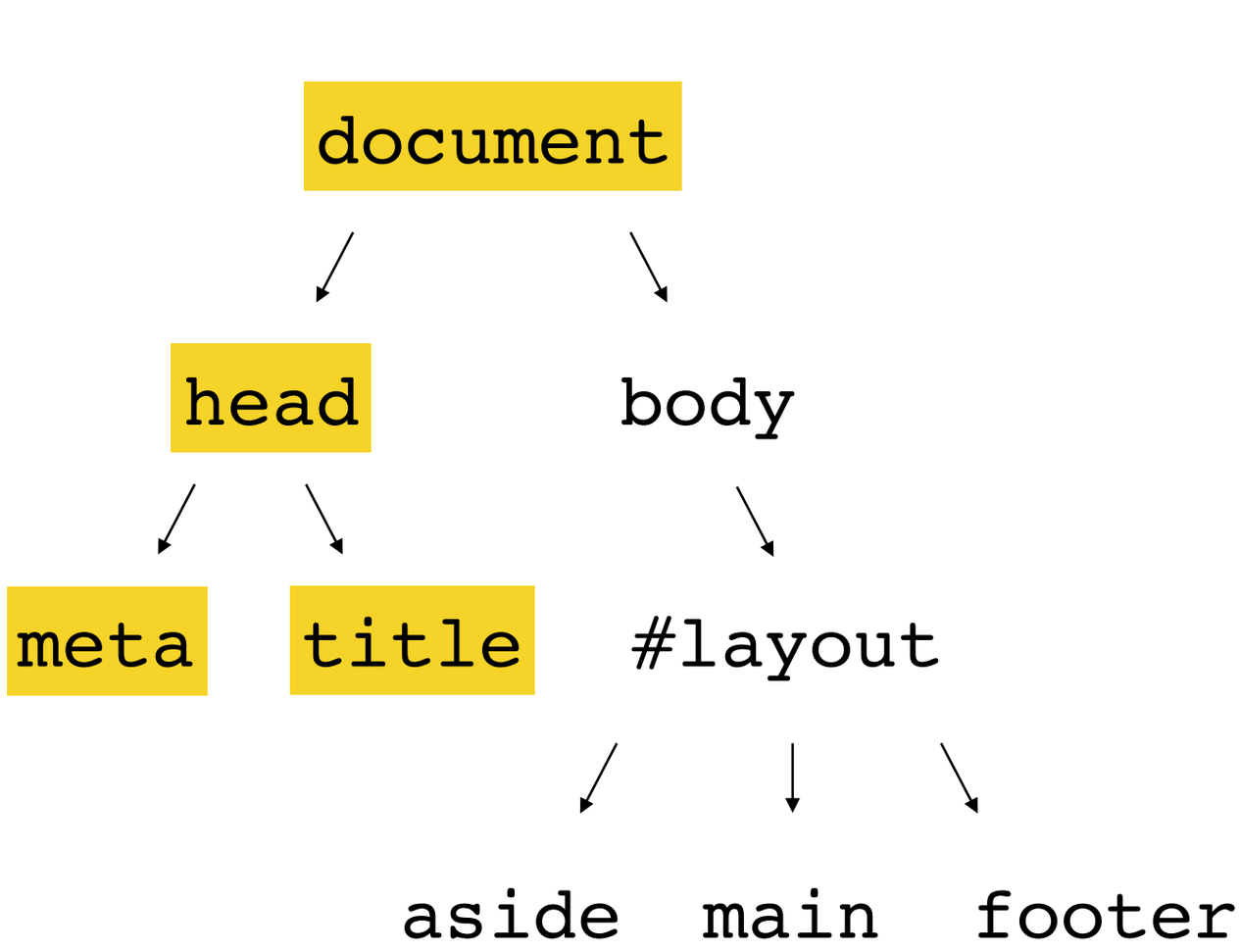
Сначала посещается узел, а потом его потомки




```
document
head
meta
```

Прямой обход в глубину

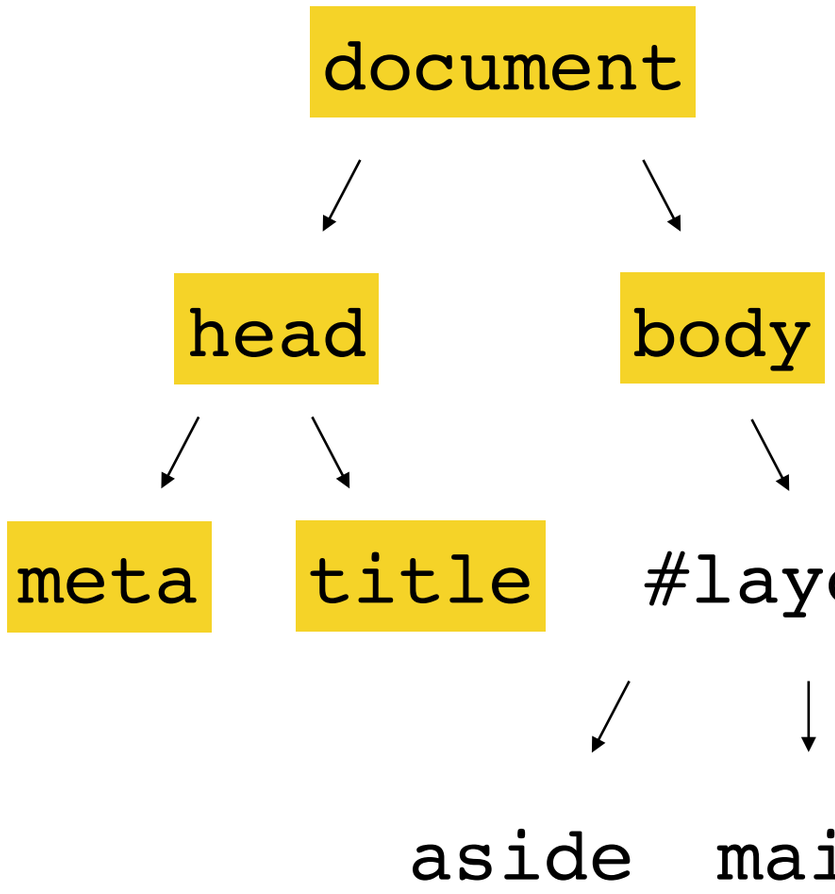
Сначала посещается узел, а потом его потомки




```
document
head
meta
title
```

Прямой обход в глубину

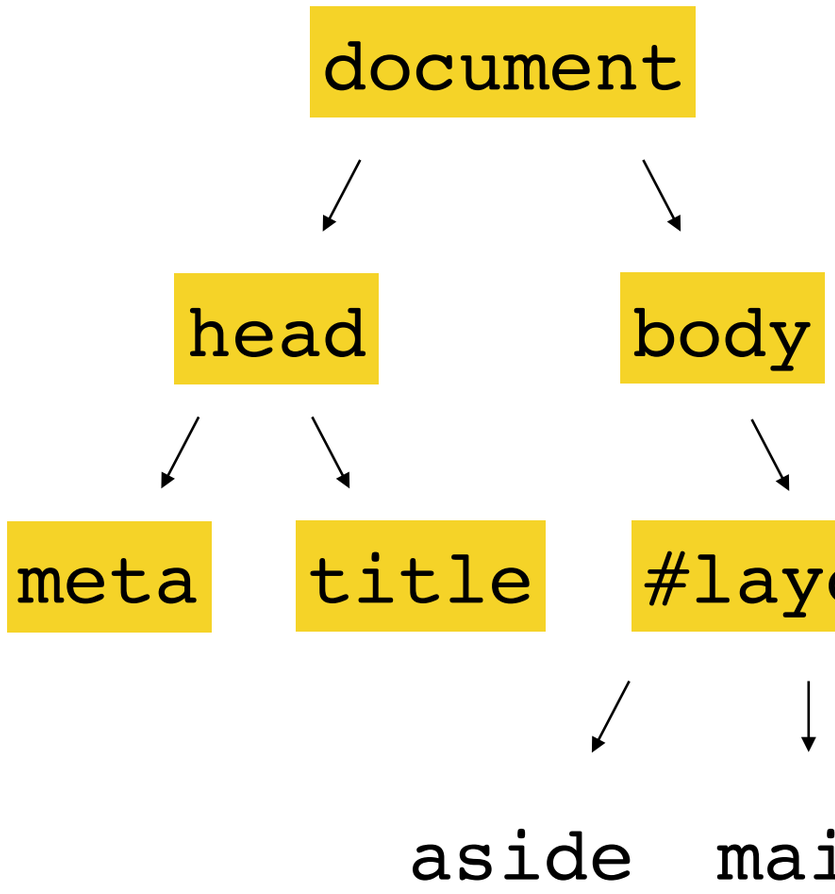
Сначала посещается узел, а потом его потомки




```
document
head
meta
title
body
```

Прямой обход в глубину

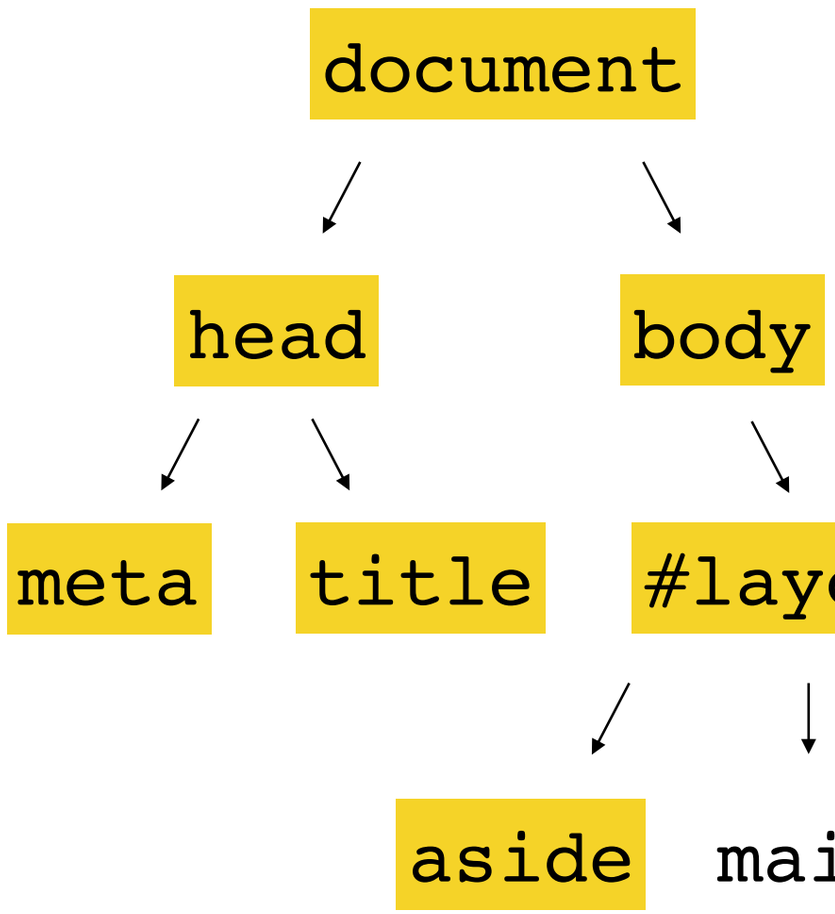
Сначала посещается узел, а потом его потомки




```
document
head
meta
title
body
#layout
```

Прямой обход в глубину

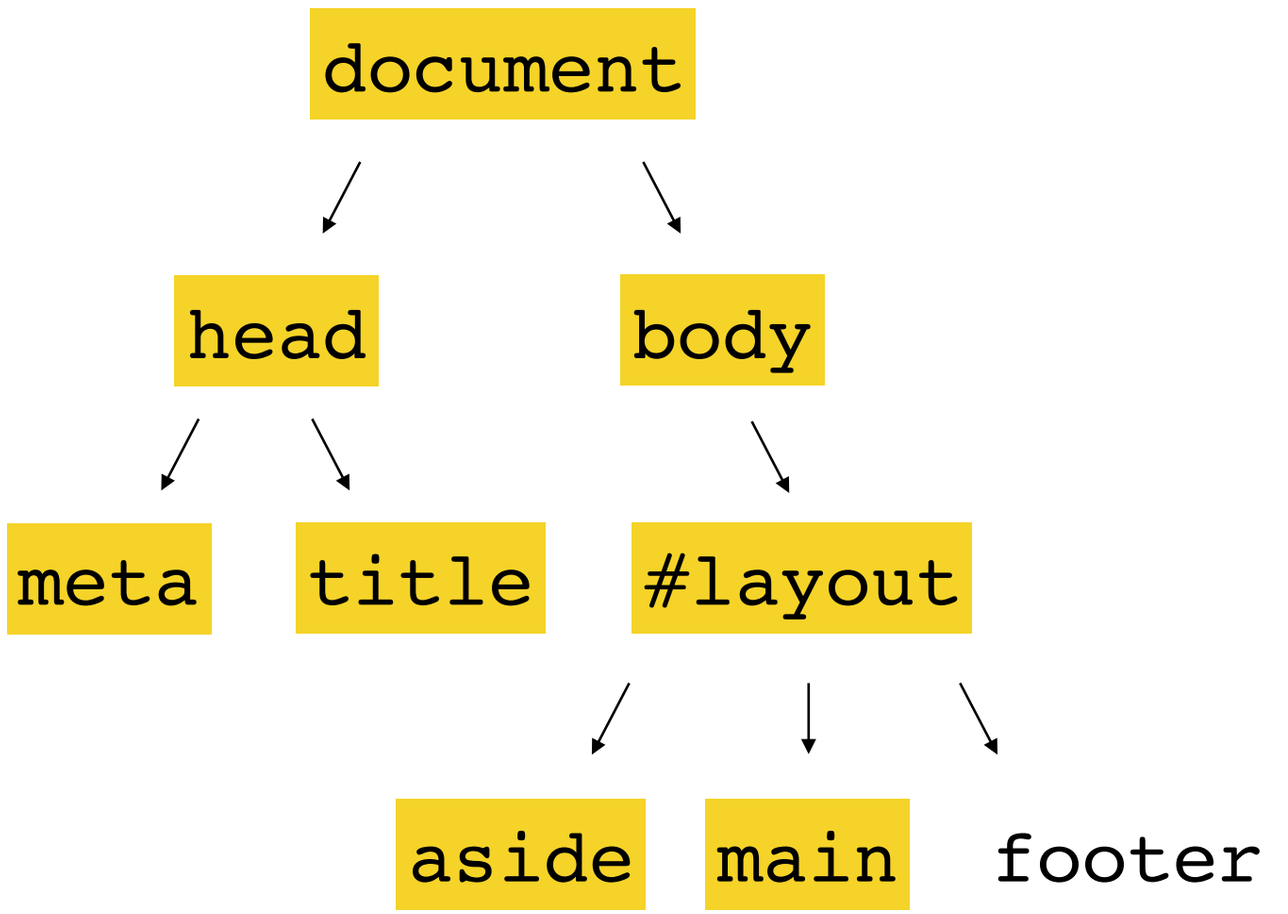
Сначала посещается узел, а потом его потомки




```
document
head
meta
title
body
#layout
aside
```

Прямой обход в глубину

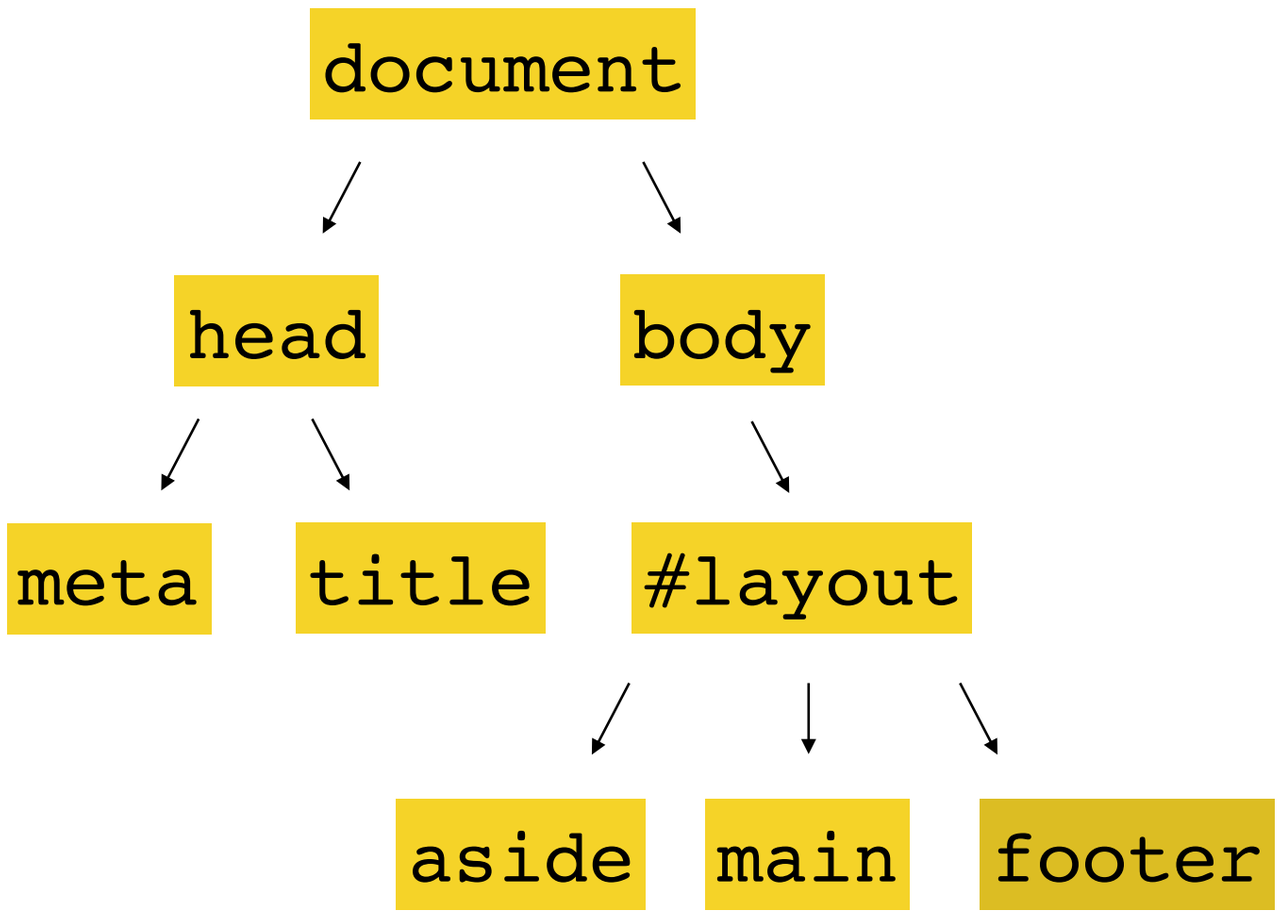
Сначала посещается узел, а потом его потомки




```
document
head
meta
title
body
#layout
aside
main
```

Прямой обход в глубину

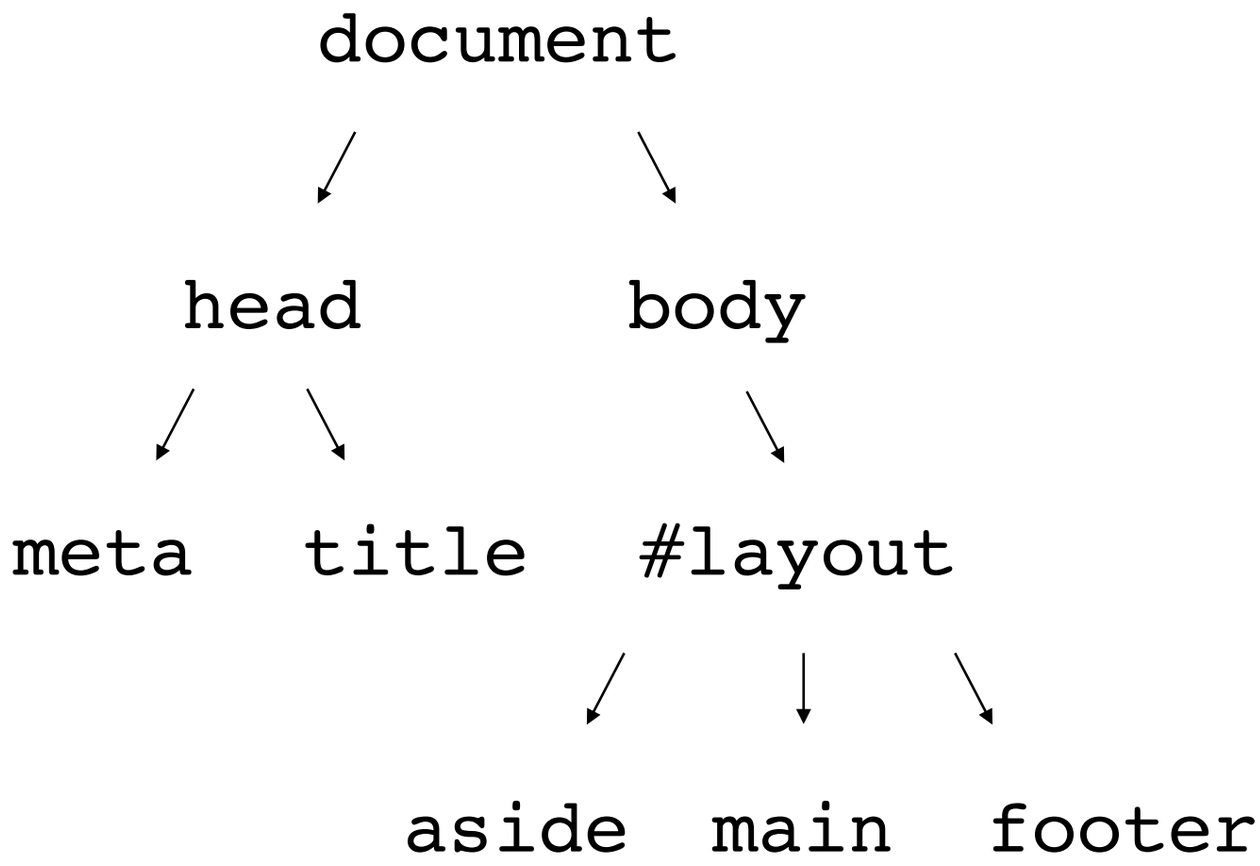
Сначала посещается узел, а потом его потомки




```
document
head
meta
title
body
#layout
aside
main
footer
```

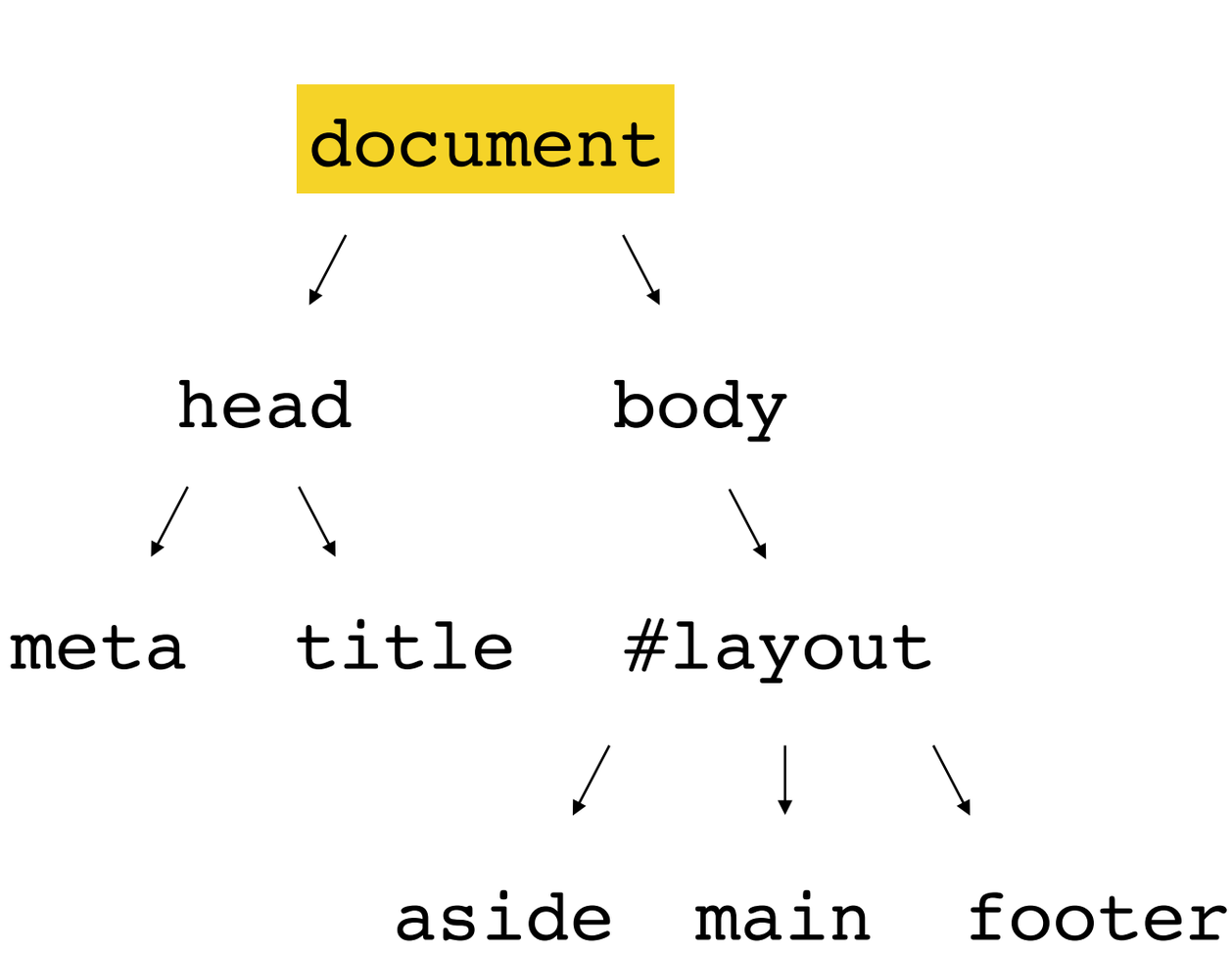
Обход в ширину

Сначала посещаются все узлы на уровне, потом их потомки и так до самого низа



Обход в ширину

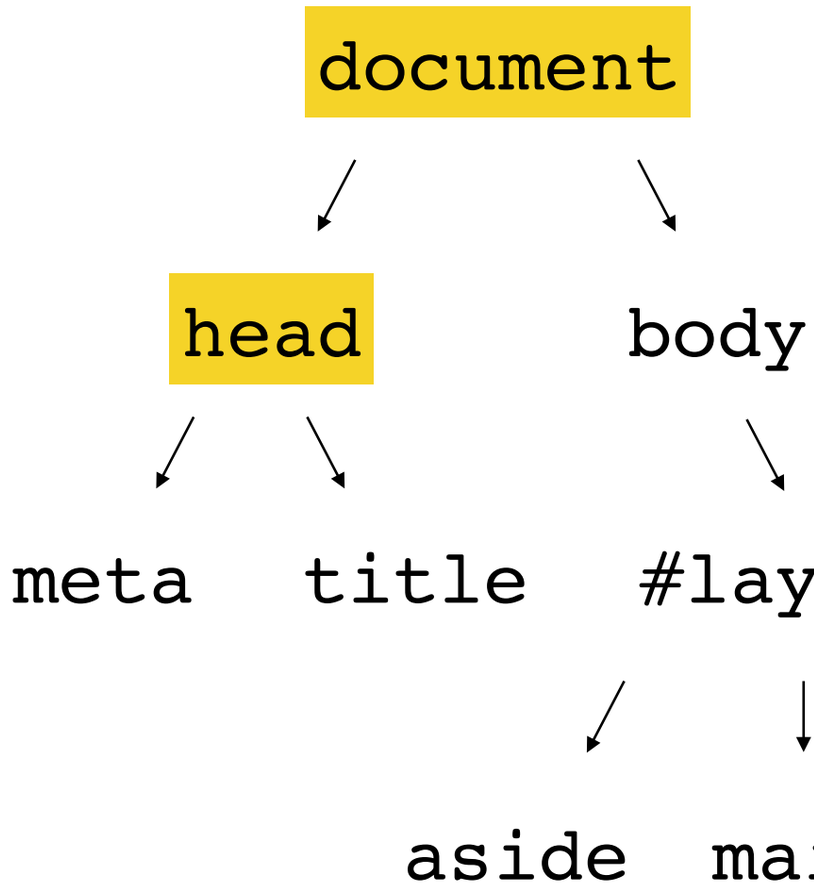
Сначала посещаются все узлы на уровне, потом их потомки и так до самого низа



document

Обход в ширину

Сначала посещаются все узлы на уровне, потом их потомки и так до самого низа

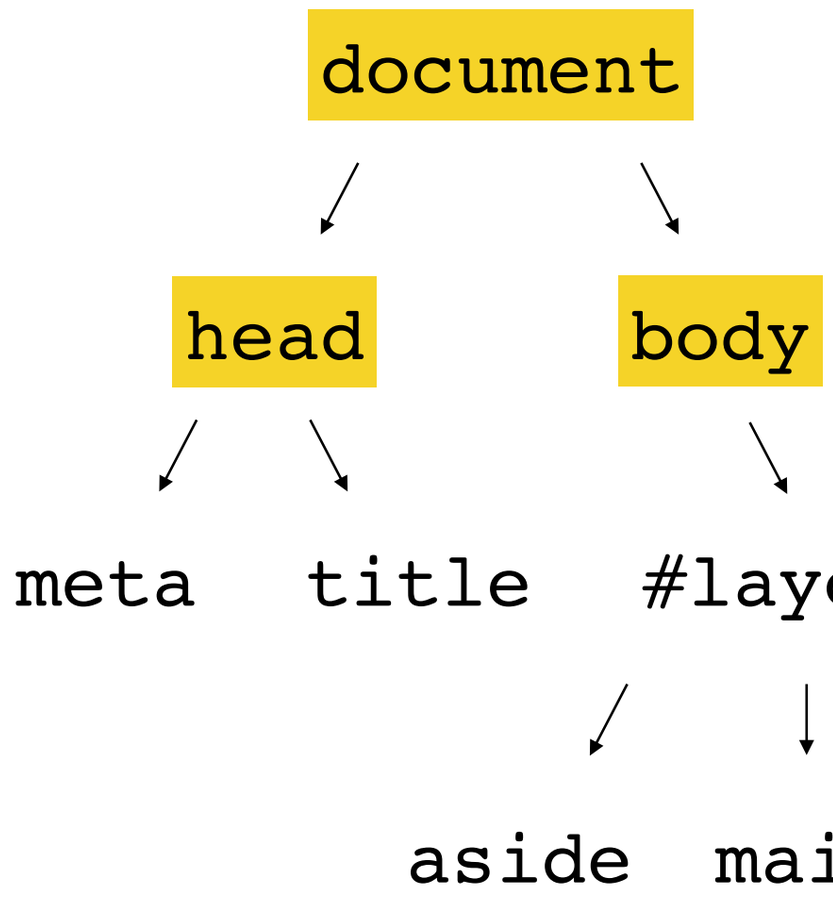


document

head

Обход в ширину

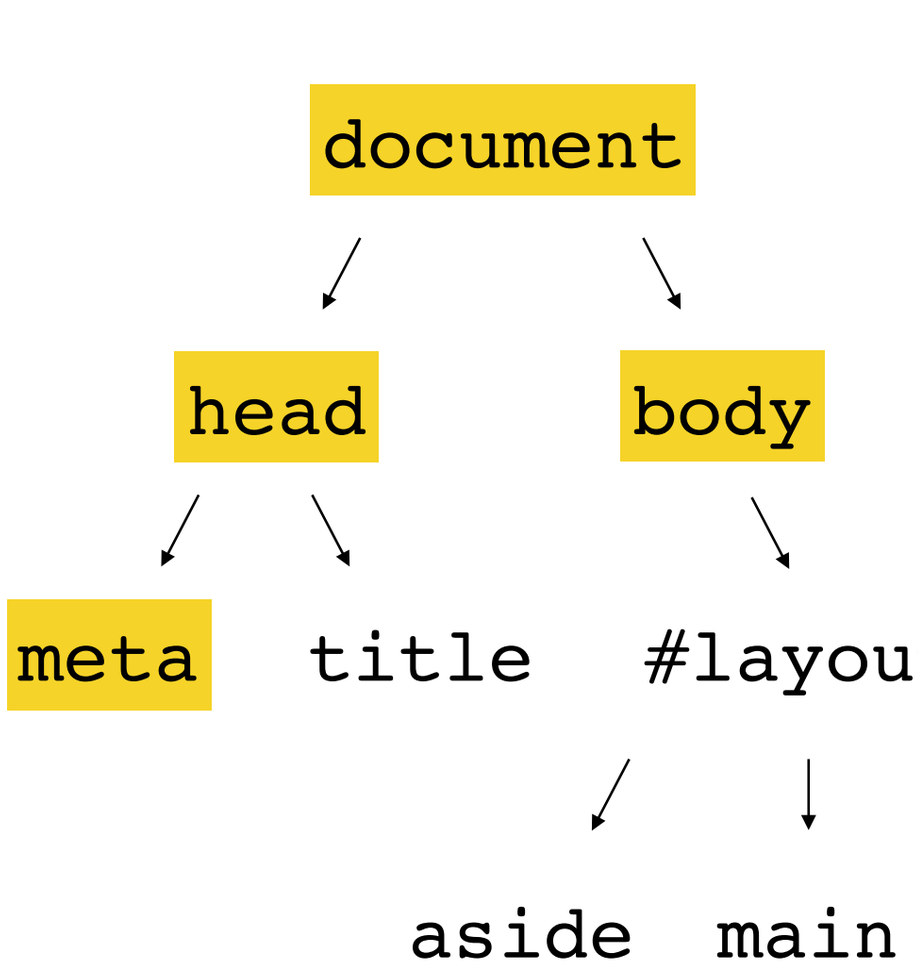
Сначала посещаются все узлы на уровне, потом их потомки и так до самого низа




```
document
head
body
```

Обход в ширину

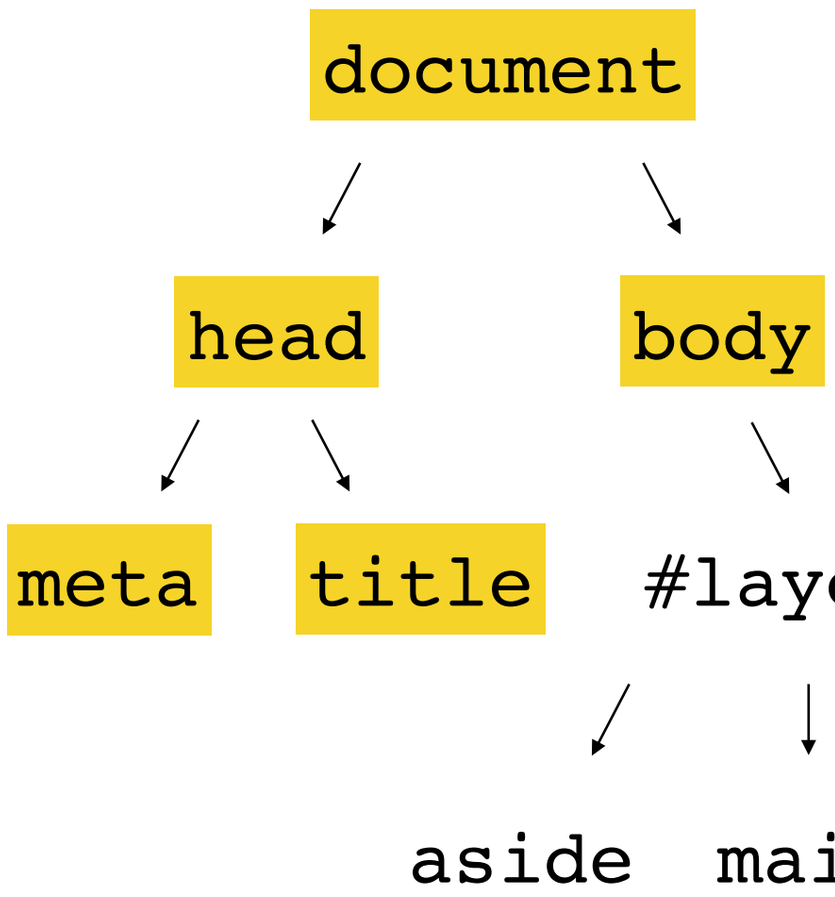
Сначала посещаются все узлы на уровне, потом их потомки и так до самого низа




```
document
head
body
meta
```

Обход в ширину

Сначала посещаются все узлы на уровне, потом их потомки и так до самого низа



document

head

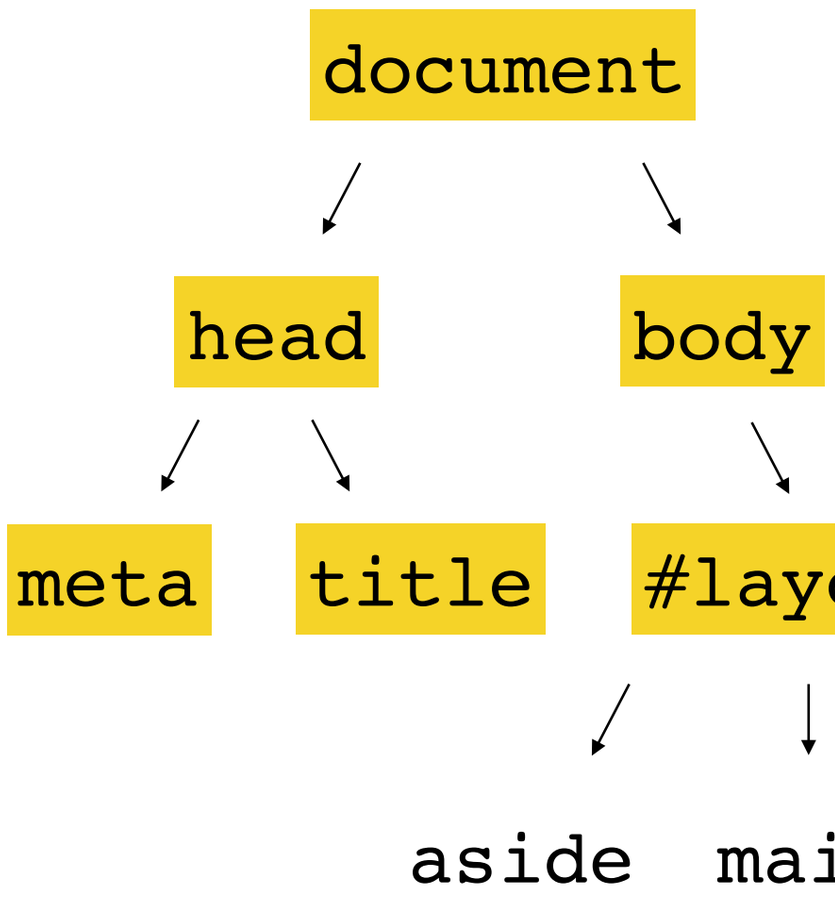
body

meta

title

Обход в ширину

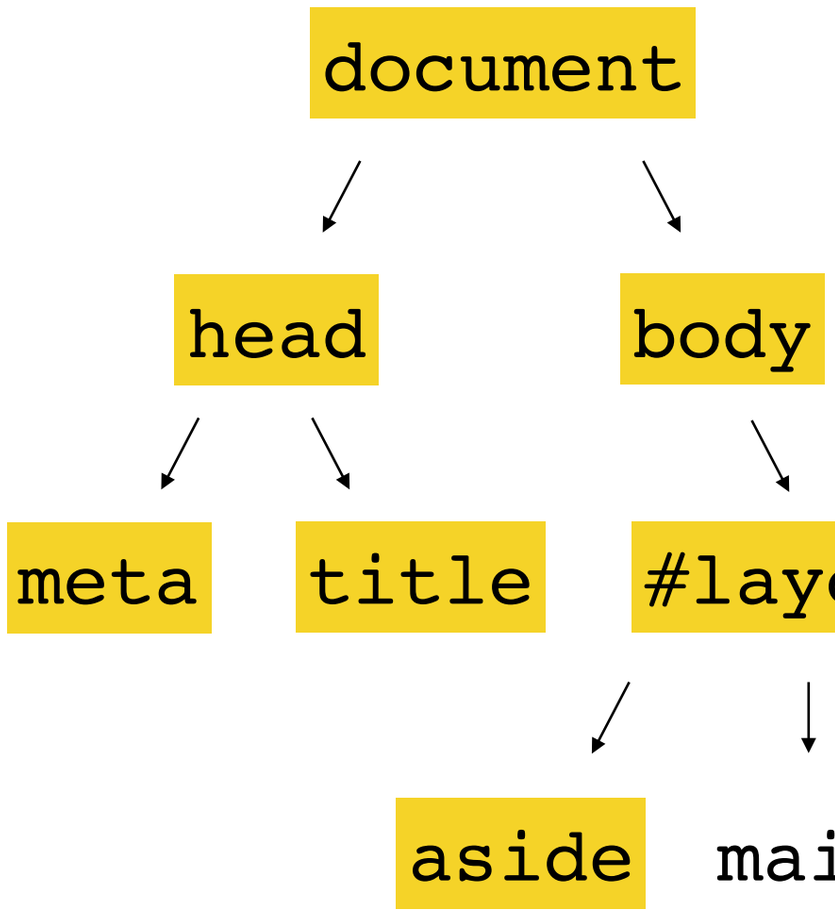
Сначала посещаются все узлы на уровне, потом их потомки и так до самого низа




```
document
head
body
meta
title
#layout
```

Обход в ширину

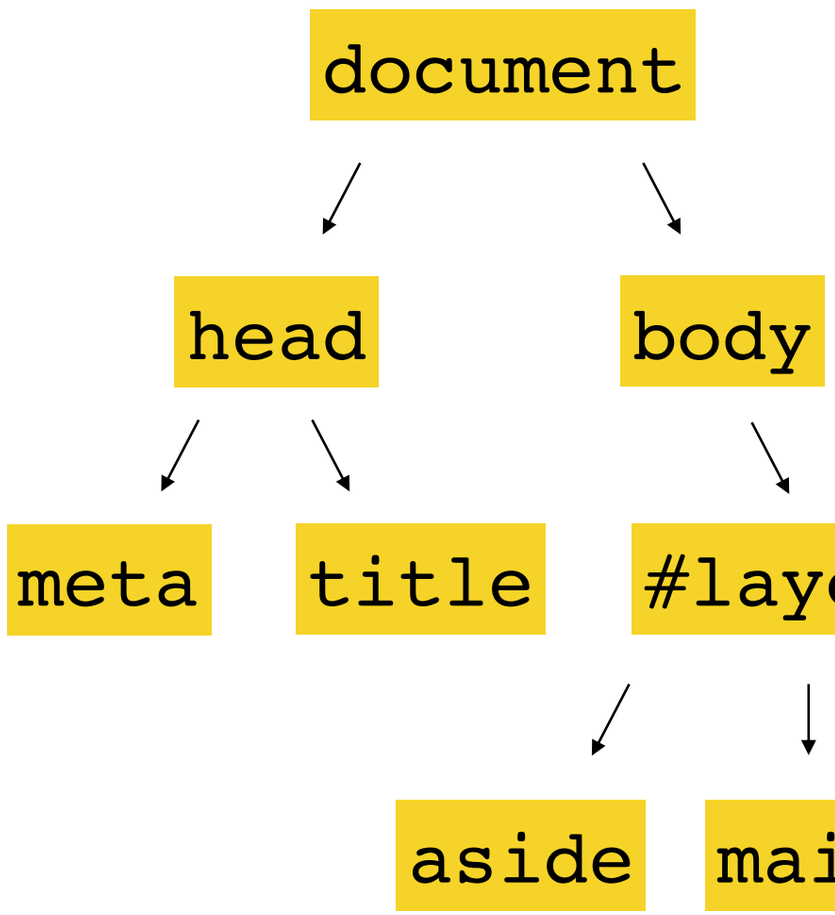
Сначала посещаются все узлы на уровне, потом их потомки и так до самого низа




```
document
head
body
meta
title
#layout
aside
```

Обход в ширину

Сначала посещаются все узлы на уровне, потом их потомки и так до самого низа



document

head

body

meta

title

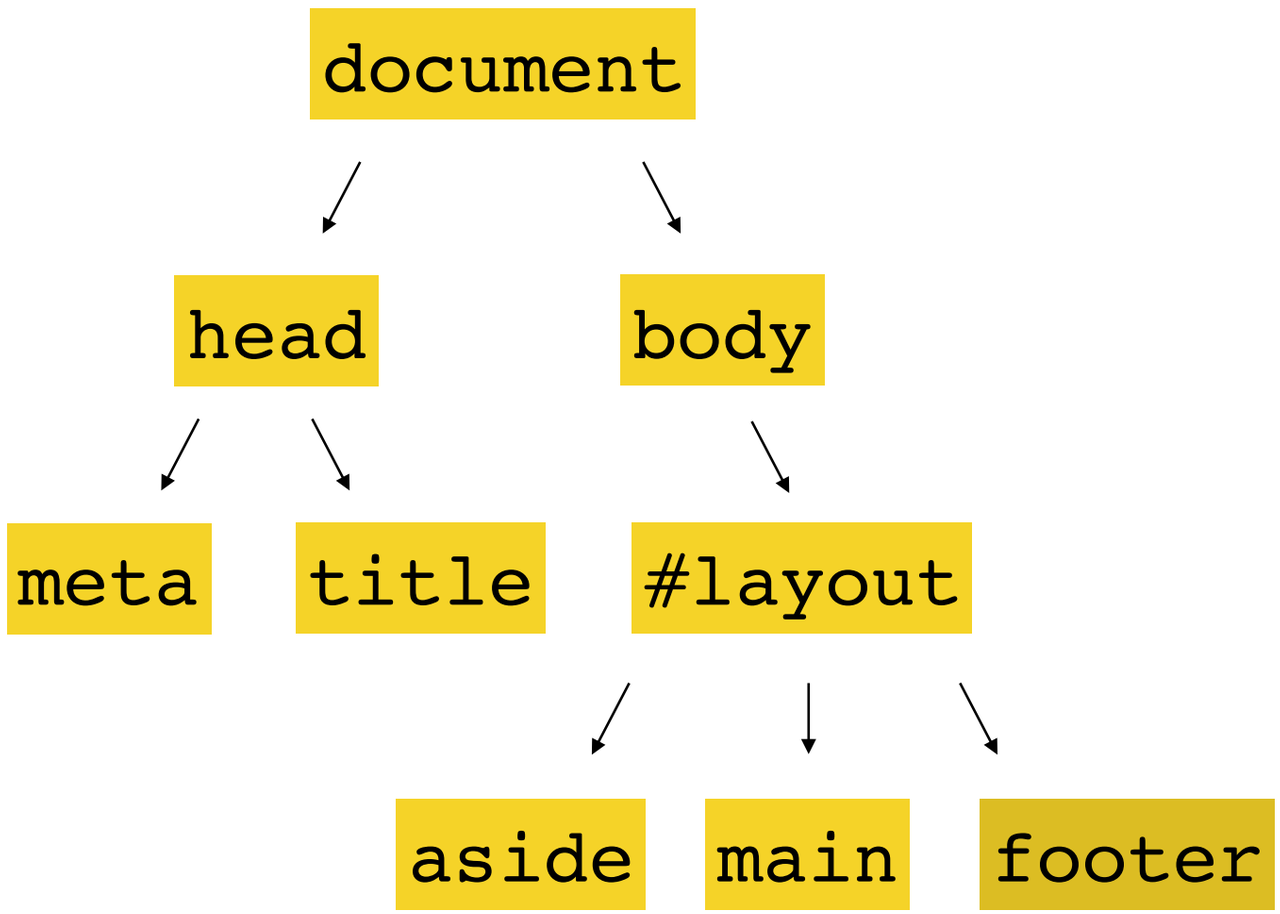
#layout

aside

main

Обход в ширину

Сначала посещаются все узлы на уровне, потом их потомки и так до самого низа




```
document
head
body
meta
title
#layout
aside
main
footer
```

Demo

Игорь

localhost:8080

HTML

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
</head>
<body>
  <div id="layout"></div>
</body>
</html>
```

CSS | LESS

```
@color: white;
@height: 100px;
@min-height: calc(@height / 2);
```

JavaScript

```
body.onload = function() {
  console.log('jopa');
}
```

×1 ×0.5 ×0.75 ×1.5 400px 800px 1000px 1500px

Demo

Игорь

localhost:8080

HTML

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
</head>
<body>
  <div id="layout"></div>
</body>
</html>
```

CSS | LESS

```
@color: white;
@height: 100px;
@min-height: calc(@height / 2);
```

JavaScript

```
body.onload = function() {
  console.log('jopa');
}
```

×1 ×0.5 ×0.75 ×1.5 400px 800px 1000px 1500px

Demo

Игорь

localhost:8080

HTML

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
</head>
<body>
  <div id="layout"></div>
</body>
</html>
```

CSS | LESS

```
@color: white;
@height: 100px;
@min-height: calc(@height / 2);
```

JavaScript

```
body.onload = function() {
  console.log('jopa');
}
```

×1 ×0.5 ×0.75 ×1.5 400px 800px 1000px 1500px 1

Demo

Игорь

localhost:8080

HTML

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
</head>
<body>
  <div id="layout"></div>
</body>
</html>
```

CSS | LESS

```
@color: white;
@height: 100px;
@min-height: calc(@height / 2);
```

JavaScript

```
body.onload = function() {
  console.log('jopa');
}
```

×1 | ×0.5 | ×0.75 | ×1.5 | 400px | 800px | 1000px | 1500px | 1

Demo

Игорь

localhost:8080

HTML

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
</head>
<body>
  <div id="layout"></div>
</body>
</html>
```

CSS | LESS

```
@color: white;
@height: 100px;
@min-height: calc(@height / 2);
```

JavaScript

```
body.onload = function() {
  console.log('jopa');
}
```

×1 | ×0.5 | ×0.75 | ×1.5 | 400px | 800px | 1000px | 1500px | 1

Demo

Игорь

localhost:8080

HTML

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
</head>
<body>
  <div id="layout"></div>
</body>
</html>
```

CSS | LESS

```
@color: white;
@height: 100px;
@min-height: calc(@height / 2);
```

JavaScript

```
body.onload = function() {
  console.log('jopa');
}
```

×1 | ×0.5 | ×0.75 | ×1.5 | 400px | 800px | 1000px | 1500px | 1

Виды нод в демке

Виды нод в демке

- **вертикальный контейнер**
  группирующий элемент, в котором все элементы располагаются один под другим

Виды нод в демке

- **вертикальный контейнер**
  группирующий элемент, в котором все элементы располагаются один под другим

- **горизонтальный контейнер**
  группирующий элемент, в котором элементы располагаются друг за другом

Виды нод в демке

- **вертикальный контейнер**
  группирующий элемент, в котором все элементы располагаются один под другим

- **горизонтальный контейнер**
  группирующий элемент, в котором элементы располагаются друг за другом

- панель с табами

Виды нод в демке

- **вертикальный контейнер**
  группирующий элемент, в котором все элементы располагаются один под другим

- **горизонтальный контейнер**
  группирующий элемент, в котором элементы располагаются друг за другом

- панель с табами

- браузер

main.js — /Users/igoralexeenko/dev/sandbox

main.js

```
1   'use strict';
2
3   const NodeType = {
4     AXIS_Y: 0,
5     AXIS_X: 1,
6     TABBAR: 2,
7     BROWSER: 3
8   };
9
```

main.js — /Users/igoralexeenko/dev/sandbox

main.js

```
1   'use strict';
2
3   const NodeType = {
4     AXIS_Y: 0,
5     AXIS_X: 1,
6     TABBAR: 2,
7     BROWSER: 3
8   };
9
10  const layout = {
11
12  };
13
```

main.js — /Users/igoralexeenko/dev/sandbox

main.js

```
1   'use strict';
2
3   const NodeType = {
4     AXIS_Y: 0,
5     AXIS_X: 1,
6     TABBAR: 2,
7     BROWSER: 3
8   };
9
10  const layout = {
11    nodeType: NodeType.AXIS_X,
12    children: []
13  };
14
```

Line 0 File 0 Project 0 ✔ No Issues main.js 12:14 LF UTF-8 JavaScript

main.js — /Users/igoralexeenko/dev/sandbox

main.js

```
'use strict';

const NodeType = {
  AXIS_Y: 0,
  AXIS_X: 1,
  TABBAR: 2,
  BROWSER: 3
};

const layout = {
  nodeType: NodeType.AXIS_X,
  children: [
    {
      nodeType: NodeType.AXIS_Y,
      children: []
    },
    {
      nodeType: NodeType.BROWSER
    }
  ]
};

```

Line 0 File 0 Project 0 ✔ No Issues main.js 19:6 LF UTF-8 JavaScript

main.js — /Users/igoralexeenko/dev/sandbox

main.js

```
'use strict';

const NodeType = {
  AXIS_Y: 0,
  AXIS_X: 1,
  TABBAR: 2,
  BROWSER: 3
};

const layout = {
  nodeType: NodeType.AXIS_X,
  children: [
    {
      nodeType: NodeType.AXIS_Y,
      children: [

      ]
    },
    {
      nodeType: NodeType.BROWSER
    }
  ]
};

```

Line 0 File 0 Project 0 ✔ No Issues main.js 16:9 LF UTF-8 JavaScript

main.js

```
'use strict';

const NodeType = {
  AXIS_Y: 0,
  AXIS_X: 1,
  TABBAR: 2,
  BROWSER: 3
};

const layout = {
  nodeType: NodeType.AXIS_X,
  children: [
    {
      nodeType: NodeType.AXIS_Y,
      children: [
        {
          nodeType: NodeType.TABBAR,
          children: ['HTML']
        },
      ]
    },
    {
      nodeType: NodeType.BROWSER
    }
  ]
};

```

main.js

```
'use strict';

const NodeType = {
  AXIS_Y: 0,
  AXIS_X: 1,
  TABBAR: 2,
  BROWSER: 3
};

const layout = {
  nodeType: NodeType.AXIS_X,
  children: [
    {
      nodeType: NodeType.AXIS_Y,
      children: [
        {
          nodeType: NodeType.TABBAR,
          children: ['HTML']
        },
        {
          nodeType: NodeType.TABBAR,
          children: ['CSS', 'LESS']
        },
        {
          nodeType: NodeType.TABBAR,
          children: ['JavaScript']
        }
      ]
    },
    {
      nodeType: NodeType.BROWSER
    }
  ]
```

main.js — /Users/igoralexeenko/dev/sandbox

main.js

```
'use strict';

const NodeType = {
  AXIS_Y: 0,
  AXIS_X: 1,
  TABBAR: 2,
  BROWSER: 3
};

const NodeTypeToReactComponent = new Map([
  [NodeType.AXIS_Y, AxisY],
  [NodeType.AXIS_X, AxisX],
  [NodeType.TABBAR, Tabs],
  [NodeType.BROWSER, IFrame],
]);

const layout = {
  nodeType: NodeType.AXIS_X,
  children: [
    {
      nodeType: NodeType.AXIS_Y,
      children: [
        {
          nodeType: NodeType.TABBAR,
          children: ['HTML']
        },
        {
          nodeType: NodeType.TABBAR,
          children: ['CSS', 'LESS']
        },
        {
          nodeType: NodeType.TABBAR,
          children: ['JavaScript']
```

Line 0 File 0 Project 0 ✔ No Issues main.js* 10:1 (7, 160) LF UTF-8 JavaScript

main.js

```
18      nodeType: NodeType.AXIS_X,
19      children: [
20        {
21          nodeType: NodeType.AXIS_Y,
22          children: [
23            {
24              nodeType: NodeType.TABBAR,
25              children: ['HTML']
26            },
27            {
28              nodeType: NodeType.TABBAR,
29              children: ['CSS', 'LESS']
30            },
31            {
32              nodeType: NodeType.TABBAR,
33              children: ['JavaScript']
34            }
35          ]
36        },
37        {
38          nodeType: NodeType.BROWSER
39        }
40      ]
41    };
42
43    let preorder = (node) => {
44      return React.createElement(
45        NodeTypeToReactComponent.get(node.nodeType), {},
46        Array.isArray(node.children) ? node.children.map((child) => preorder(child)) : null
47      );
48    };
49
```

main.js

```
20        {
21          nodeType: NodeType.AXIS_Y,
22          children: [
23            {
24              nodeType: NodeType.TABBAR,
25              children: ['HTML']
26            },
27            {
28              nodeType: NodeType.TABBAR,
29              children: ['CSS', 'LESS']
30            },
31            {
32              nodeType: NodeType.TABBAR,
33              children: ['JavaScript']
34            }
35          ]
36        },
37        {
38          nodeType: NodeType.BROWSER
39        }
40      ]
41    };
42
43    let preorder = (node) => {
44      return React.createElement(
45        NodeTypeToReactComponent.get(node.nodeType), {},
46        Array.isArray(node.children) ? node.children.map((child) => preorder(child)) : null
47      );
48    };
49
50    React.render(preorder(layout), document.querySelector('.layout'));
51
```

main.js

```
22          children: [
23            {
24              nodeType: NodeType.TABBAR,
25              children: ['HTML']
26            },
27            {
28              nodeType: NodeType.TABBAR,
29              children: ['CSS', 'LESS']
30            },
31            {
32              nodeType: NodeType.TABBAR,
33              children: ['JavaScript']
34            }
35          ]
36        },
37        {
38          nodeType: NodeType.BROWSER
39        }
40      ]
41    };
42
43    let preorder = (node) => {
44      return React.createElement(
45        NodeTypeToReactComponent.get(node.nodeType), {},
46        Array.isArray(node.children) ? node.children.map((child) => preorder(child)) : null
47      );
48    };
49
50    React.render(preorder(layout), document.querySelector('.layout'));
51
52    localStorage.set('layout', JSON.stringify(layout));
53
```

Demo

Игорь

localhost:8080

HTML

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
</head>
<body>
  <div id="layout"></div>
</body>
</html>
```

CSS | LESS

```
@color: white;
@height: 100px;
@min-height: calc(@height / 2);
```

JavaScript

```
body.onload = function() {
  console.log('jopa');
}
```

×1 ×0.5 ×0.75 ×1.5 400px 800px 1000px 1500px

Demo

Игорь

localhost:8080

HTML

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
</head>
<body>
  <div id="layout"></div>
</body>
</html>
```

CSS | LESS

```
@color: white;
@height: 100px;
@min-height: calc(@height / 2);
```

JavaScript

```
body.onload = function() {
  console.log('jopa');
}
```

×1 ×0.5 ×0.75 ×1.5 400px 800px 1000px 1500px

Demo

Игорь

localhost:8080

HTML

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
</head>
<body>
  <div id="layout"></div>
</body>
</html>
```

CSS | LESS

```
@color: white;
@height: 100px;
@min-height: calc(@height / 2);
```

JavaScript

```
body.onload = function() {
  console.log('jopa');
}
```

×1 | ×0.5 | ×0.75 | ×1.5 | 400px | 800px | 1000px | 1500px | 1

Demo

Игорь

localhost:8080

HTML

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
</head>
<body>
  <div id="layout"></div>
</body>
</html>
```

CSS | LESS

```
@color: white;
@height: 100px;
@min-height: calc(@height / 2);
```

JavaScript

```
body.onload = function() {
  console.log('jopa');
}
```

×1 | ×0.5 | ×0.75 | ×1.5 | 400px | 800px | 1000px | 1500px | 1

Demo

Игорь

localhost:8080

HTML

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
</head>
<body>
  <div id="layout"></div>
</body>
</html>
```

CSS | LESS

```
@color: white;
@height: 100px;
@min-height: calc(@height / 2);
```

JavaScript

```
body.onload = function() {
  console.log('jopa');
}
```

×1 | ×0.5 | ×0.75 | ×1.5 | 400px | 800px | 1000px | 1500px | 1

Demo

Игорь

localhost:8080

HTML

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
</head>
<body>
  <div id="layout"></div>
</body>
</html>
```

CSS | LESS

```
@color: white;
@height: 100px;
@min-height: calc(@height / 2);
```

JavaScript

```
body.onload = function() {
  console.log('jopa');
}
```

×1 | ×0.5 | ×0.75 | ×1.5 | 400px | 800px | 1000px | 1500px | 1

Demo

Игорь

localhost:8080

HTML

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
</head>
<body>
  <div id="layout"></div>
</body>
</html>
```

CSS | LESS

```
@color: white;
@height: 100px;
@min-height: calc(@height / 2);
```

JavaScript

```
body.onload = function() {
  console.log('jopa');
}
```

×1 | ×0.5 | ×0.75 | ×1.5 | 400px | 800px | 1000px | 1500px | 1

Demo

Игорь

localhost:8080

HTML

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
</head>
<body>
  <div id="layout"></div>
</body>
</html>
```

CSS | LESS

```
@color: white;
@height: 100px;
@min-height: calc(@height / 2);
```

JavaScript

```
body.onload = function() {
  console.log('jopa');
}
```

×1 | ×0.5 | ×0.75 | ×1.5 | 400px | 800px | 1000px | 1500px | 1

Demo

Игорь

localhost:8080

HTML

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
</head>
<body>
  <div id="layout"></div>
</body>
</html>
```

CSS | LESS

```
@color: white;
@height: 100px;
@min-height: calc(@height / 2);
```

JavaScript

```
body.onload = function() {
  console.log('jopa');
}
```

×1 | ×0.5 | ×0.75 | ×1.5 | 400px | 800px | 1000px | 1500px

Для оптимизации скорости отрисовки можно вместо поиска в глубину запустить поиск в ширину, чтобы сначала появлялись контейнеры и только потом их содержимое

(не в случае Реакта, в его контейнер фиг дорисуешь)

Demo

Игорь

localhost:8080

HTML

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
</head>
<body>
  <div id="layout"></div>
</body>
</html>
```

CSS | LESS

```
@color: white;
@height: 100px;
@min-height: calc(@height / 2);
```

JavaScript

```
body.onload = function() {
  console.log('jopa');
}
```

×1 ×0.5 ×0.75 ×1.5 400px 800px 1000px 1500px

Demo

Игорь

localhost:8080

HTML

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
</head>
<body>
  <div id="layout"></div>
</body>
</html>
```

CSS | LESS

```
@color: white;
@height: 100px;
@min-height: calc(@height / 2);
```

JavaScript

```
body.onload = function() {
  console.log('jopa');
}
```

×1 ×0.5 ×0.75 ×1.5 400px 800px 1000px 1500px

Demo

Игорь

localhost:8080

HTML

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
</head>
<body>
  <div id="layout"></div>
</body>
</html>
```

CSS | LESS

```
@color: white;
@height: 100px;
@min-height: calc(@height / 2);
```

JavaScript

```
body.onload = function() {
  console.log('jopa');
}
```

×1 | ×0.5 | ×0.75 | ×1.5 | 400px | 800px | 1000px | 1500px | 1

Demo

Игорь

localhost:8080

HTML

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
</head>
<body>
  <div id="layout"></div>
</body>
</html>
```

CSS | LESS

```
@color: white;
@height: 100px;
@min-height: calc(@height / 2);
```

JavaScript

```
body.onload = function() {
  console.log('jopa');
}
```

×1 ×0.5 ×0.75 ×1.5 400px 800px 1000px 1500px

Demo

Игорь

localhost:8080

HTML

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
</head>
<body>
  <div id="layout"></div>
</body>
</html>
```

CSS | LESS

```
@color: white;
@height: 100px;
@min-height: calc(@height / 2);
```

JavaScript

```
body.onload = function() {
  console.log('jopa');
}
```

×1 ×0.5 ×0.75 ×1.5 400px 800px 1000px 1500px

Demo

Игорь

localhost:8080

HTML

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
</head>
<body>
  <div id="layout"></div>
</body>
</html>
```

CSS | LESS

```
@color: white;
@height: 100px;
@min-height: calc(@height / 2);
```

JavaScript

```
body.onload = function() {
  console.log('jopa');
}
```

×1 ×0.5 ×0.75 ×1.5 400px 800px 1000px 1500px

Demo

Игорь

localhost:8080

HTML

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
</head>
<body>
  <div id="layout"></div>
</body>
</html>
```

CSS | LESS

```
@color: white;
@height: 100px;
@min-height: calc(@height / 2);
```

JavaScript

```
body.onload = function() {
  console.log('jopa');
}
```

×1 ×0.5 ×0.75 ×1.5 400px 800px 1000px 1500px

Demo

Игорь

localhost:8080

HTML

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
</head>
<body>
  <div id="layout"></div>
</body>
</html>
```

CSS | LESS

```
@color: white;
@height: 100px;
@min-height: calc(@height / 2);
```

JavaScript

```
body.onload = function() {
  console.log('jopa');
}
```

×1 | ×0.5 | ×0.75 | ×1.5 | 400px | 800px | 1000px | 1500px

Demo

Игорь

localhost:8080

HTML

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
</head>
<body>
  <div id="layout"></div>
</body>
</html>
```

CSS | LESS

```
@color: white;
@height: 100px;
@min-height: calc(@height / 2);
```

JavaScript

```
body.onload = function() {
  console.log('jopa');
}
```

×1 | ×0.5 | ×0.75 | ×1.5 | 400px | 800px | 1000px | 1500px

Одна и та же структура

Одна и та же структура

- описывает лейаут

Одна и та же структура

- описывает лейаут
- сохраняется и передается в виде данных

Одна и та же структура

- описывает лейаут
- сохраняется и передается в виде данных
- легко читается

Одна и та же структура

- описывает лейаут
- сохраняется и передается в виде данных
- легко читается
- быстро выполняется (*нативный объект JS*)

Доставка букета

- [ ] Нужна доставка
  - [ ] Вложить в букет открытку
  - [ ] Подарить мягкую игрушку
    - [ ] Оторвать ей лапу
    - [ ] Зашить ей рот красной ниткой
  - [ ] Сделать фотографию

Адрес доставки

```
 3    <head>
 4      <meta charset="utf-8">
 5      <title></title>
 6      <style media="screen">…
32      </style>
33    </head>
34    <body>
35
36    <form>
37      <div id="layout">
38        <h3>Доставка букета</h3>
39        <fieldset>
40          <input type="checkbox" id="v1" name="v1" /><label for="v1"> Нужна доставка</
41          <fieldset>
42            <input type="checkbox" id="v2" name="v2" /><label for="v2"> Вложить в буке
43            <input type="checkbox" id="v2" name="v2" /><label for="v2"> Подарить мягку
44            <fieldset>
45              <input type="checkbox" id="v4" name="v4" /><label for="v4"> Оторвать ей л
46              <input type="checkbox" id="v4" name="v4" /><label for="v4"> Зашить ей рот
47            </fieldset>
48            <input type="checkbox" id="v2" name="v2" /><label for="v2"> Сделать фотогра
49            <input type="text" value="Адрес доставки" name="v3" />
50          </fieldset>
51        </fieldset>
52      </div>
53    </form>
54
55    <script src="main.js"></script>
56
57    </body>
58    </html>
59
```

main.js — /Users/igoralexeenko/dev/sandbox

main.js | index.html

```
1   'use strict';
2
3   let deps = {
4     el: document.forms[0]['v1'],
5     children: []
6   };
7
```

Line 0 | File 0 | Project 0 | No Issues | main.js | 5:14 | LF | UTF-8 | JavaScript

main.js — /Users/igoralexeenko/dev/sandbox

main.js | index.html

```
'use strict';

let deps = {
  el: document.forms[0]['v1'],
  children: [
    { el: document.forms[0]['v2'][0] },
    {
      el: document.forms[0]['v2'][1],
      children: []
    },
    { el: document.forms[0]['v2'][2] },
    { el: document.forms[0]['v3'] }
  ]
};
```

Line 0 | File 0 | Project 0 | No Issues | main.js | 12:36 | LF | UTF-8 | JavaScript

main.js — /Users/igoralexeenko/dev/sandbox

main.js | index.html

```
'use strict';

let deps = {
  el: document.forms[0]['v1'],
  children: [
    { el: document.forms[0]['v2'][0] },
    {
      el: document.forms[0]['v2'][1],
      children: [
        { el: document.forms[0]['v4'][0] },
        { el: document.forms[0]['v4'][1] }
      ]
    },
    { el: document.forms[0]['v2'][2] },
    { el: document.forms[0]['v3'] }
  ]
};

```

Line 0 | File 0 | Project 0 | No Issues | main.js | 11:43 | LF | UTF-8 | JavaScript

```
'use strict';

let deps = {
  el: document.forms[0]['v1'],
  children: [
    { el: document.forms[0]['v2'][0] },
    {
      el: document.forms[0]['v2'][1],
      children: [
        { el: document.forms[0]['v4'][0] },
        { el: document.forms[0]['v4'][1] }
      ]
    },
    { el: document.forms[0]['v2'][2] },
    { el: document.forms[0]['v3'] }
  ]
};


let preorder = (node, parent) => {
  node.el.disabled = parent && !parent.el.checked;

  if (node.children) {
    node.children.forEach((child) => preorder(child, node));
  }
};

```

main.js — /Users/igoralexeenko/dev/sandbox

main.js | index.html

```
'use strict';

let deps = {
  el: document.forms[0]['v1'],
  children: [
    { el: document.forms[0]['v2'][0] },
    {
      el: document.forms[0]['v2'][1],
      children: [
        { el: document.forms[0]['v4'][0] },
        { el: document.forms[0]['v4'][1] }
      ]
    },
    { el: document.forms[0]['v2'][2] },
    { el: document.forms[0]['v3'] }
  ]
};


let preorder = (node, parent) => {
  node.el.disabled = parent && !parent.el.checked;

  if (node.children) {
    node.children.forEach((child) => preorder(child, node));
  }
};


preorder(deps, null);

```

Line 0 | File 0 | Project 0 | No Issues | main.js | 29:22 | LF | UTF-8 | JavaScript

```
let deps = {
  el: document.forms[0]['v1'],
  children: [
    { el: document.forms[0]['v2'][0] },
    {
      el: document.forms[0]['v2'][1],
      children: [
        { el: document.forms[0]['v4'][0] },
        { el: document.forms[0]['v4'][1] }
      ]
    },
    { el: document.forms[0]['v2'][2] },
    { el: document.forms[0]['v3'] }
  ]
};


let preorder = (node, parent) => {
  node.el.disabled = parent && !parent.el.checked;

  if (node.children) {
    node.children.forEach((child) => preorder(child, node));
  }
};


preorder(deps, null);


document.forms[0].onchange = () => {
  preorder(deps, null);
};

```

Доставка букета

- [ ] Нужна доставка
  - [ ] Вложить в букет открытку
  - [ ] Подарить мягкую игрушку
    - [ ] Оторвать ей лапу
    - [ ] Зашить ей рот красной ниткой
  - [ ] Сделать фотографию

Адрес доставки

Доставка букета

- [x] Нужна доставка
  - [ ] Вложить в букет открытку
  - [ ] Подарить мягкую игрушку
    - [ ] Оторвать ей лапу
    - [ ] Зашить ей рот красной ниткой
  - [ ] Сделать фотографию

Адрес доставки

Доставка букета

- [x] Нужна доставка
  - [ ] Вложить в букет открытку
  - [x] Подарить мягкую игрушку
    - [ ] Оторвать ей лапу
    - [ ] Зашить ей рот красной ниткой
  - [ ] Сделать фотографию

Адрес доставки

Доставка букета

- [x] Нужна доставка
  - [ ] Вложить в букет открытку
  - [x] Подарить мягкую игрушку
    - [ ] Оторвать ей лапу
    - [x] Зашить ей рот красной ниткой
  - [ ] Сделать фотографию

Адрес доставки

Доставка букета

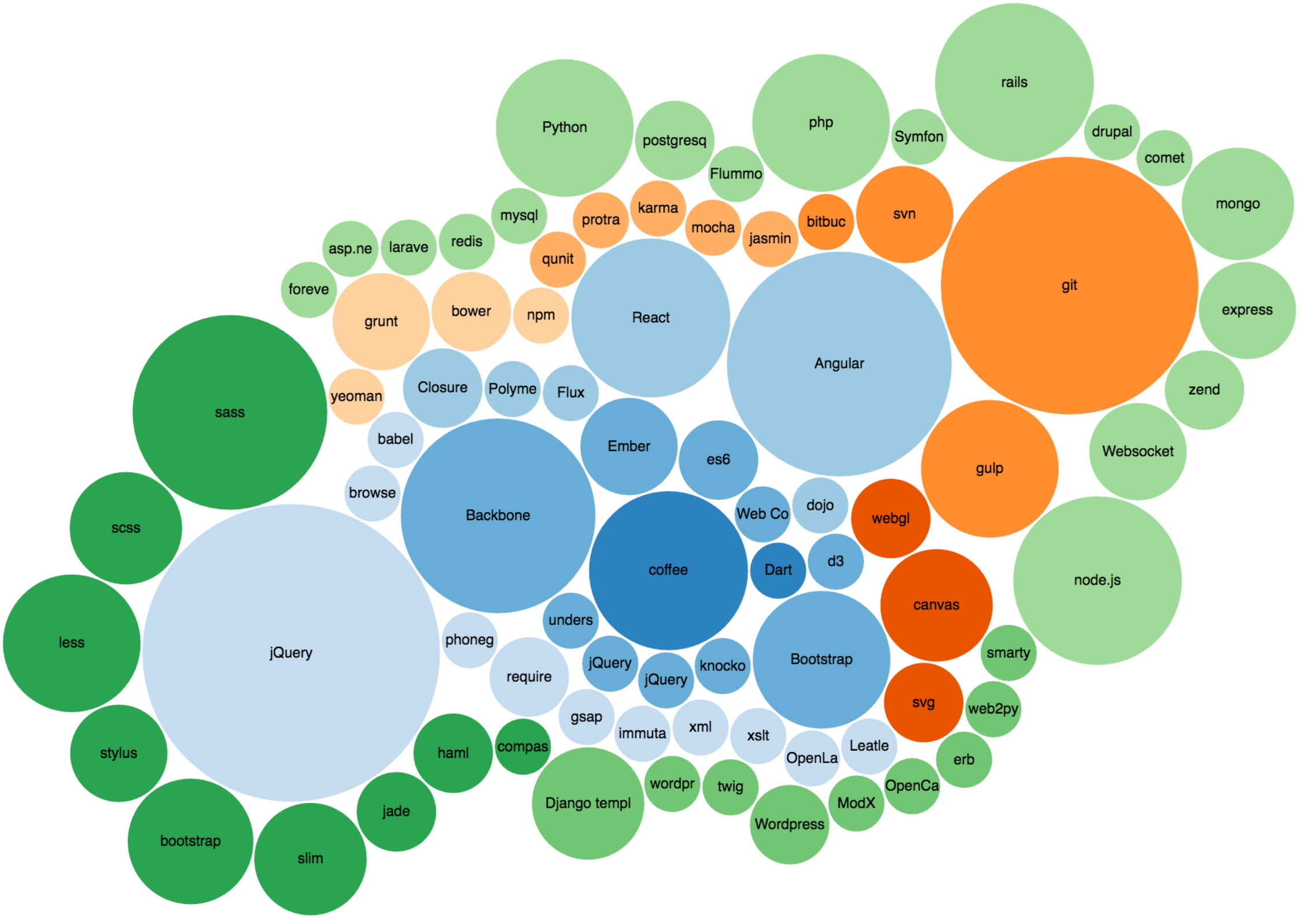
- [x] Нужна доставка
  - [ ] Вложить в букет открытку
  - [x] Подарить мягкую игрушку
    - [ ] Оторвать ей лапу
    - [x] Зашить ей рот красной ниткой
  - [ ] Сделать фотографию

Адрес доставки

Статистика требований к разработчикам на основе вакансий Моего Круга с помощью библиотеки d3.js

main.js

```
{
  "name": "data",
  "children": [
    {
      "name": "Client side",
      "children": [
        {
          "name": "Language",
          "children": [
            { "name": "coffee", "size": 8 },
            { "name": "Dart", "size": 1 }
          ]
        },
        {
          "name": "Framework / tool",
          "children": [
        {
          "name": "Libraries",
          "children": [
            { "name": "Web Components", "size":1 },
            { "name": "es6", "size": 2 },
            { "name": "Ember", "size": 3 },
            { "name": "Backbone", "size": 12 },
            { "name": "underscore", "size": 1 },
            { "name": "jQuery UI", "size":1 },
            { "name": "jQuery mobile", "size":1 },
            { "name": "knockout", "size": 1 },
            { "name": "Bootstrap", "size": 6 },
            { "name": "d3", "size": 1 }
          ]
        },
        {
          "name": "Framework",
```


Python
postgresq
Flummo
php
Symfon
rails
drupal
comet
mongo
mysql
protra
karma
mocha
jasmin
bitbuc
svn
git
express
asp.ne
larave
redis
qunit
foreve
grunt
bower
npm
React
Angular
zend
sass
yeoman
Closure
Polyme
Flux
babel
Ember
es6
gulp
Websocket
browse
Backbone
Web Co
dojo
webgl
scss
coffee
Dart
d3
node.js
canvas
less
jQuery
unders
phoneg
require
jQuery
jQuery
knocko
Bootstrap
smarty
svg
web2py
gsap
immuta
xml
xslt
OpenLa
Leatle
erb
stylus
haml
compas
Django templ
wordpr
twig
ModX
OpenCa
bootstrap
jade
slim
Wordpress

– Спасибо!

Я в соцсетях: iamo0

Айда наставничать в Академию:
https://htmlacademy.ru/intensive/javascript/tutors

Примеры:
https://gist.github.com/o0/