Яндекс Карты

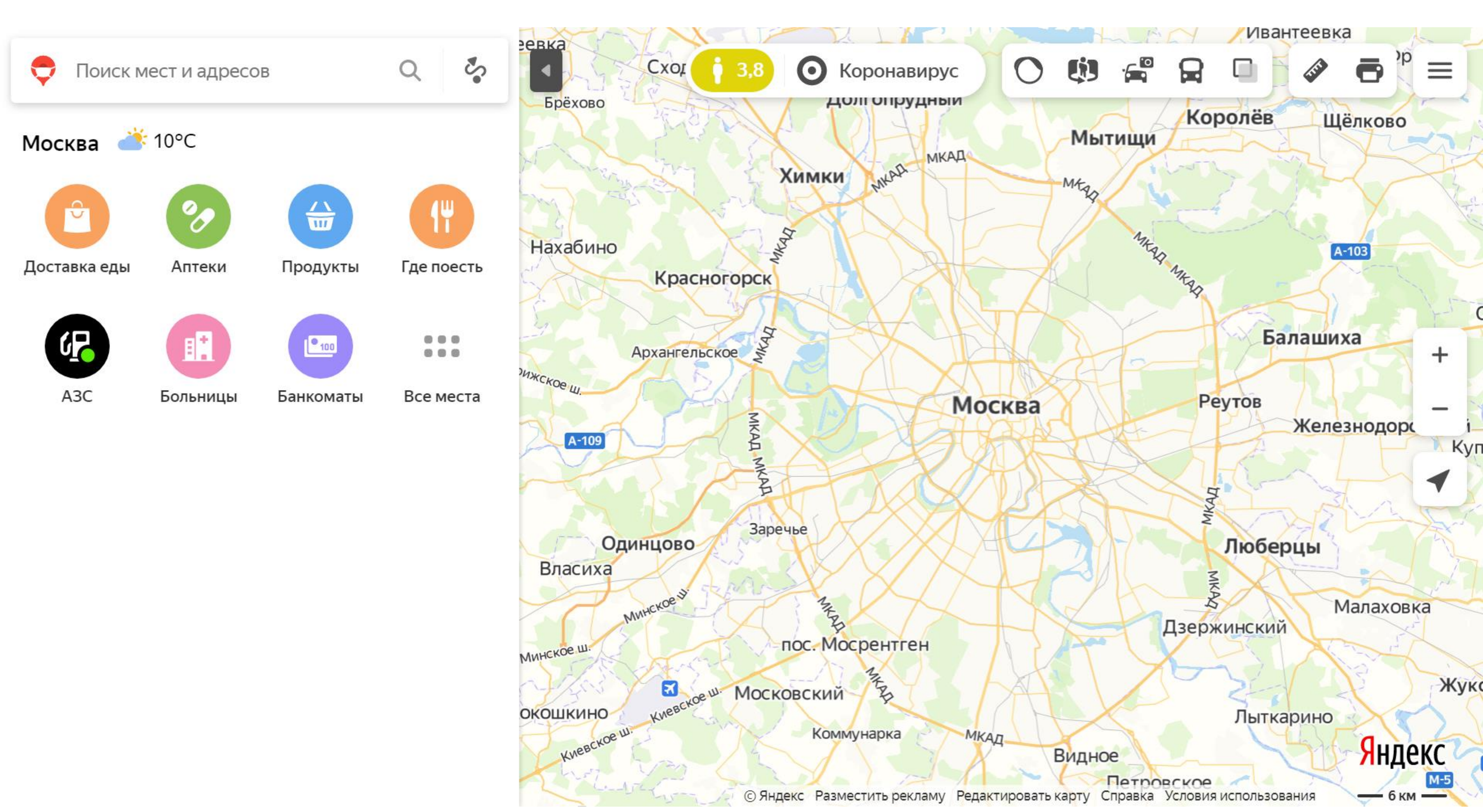
WebAssembly без купюр

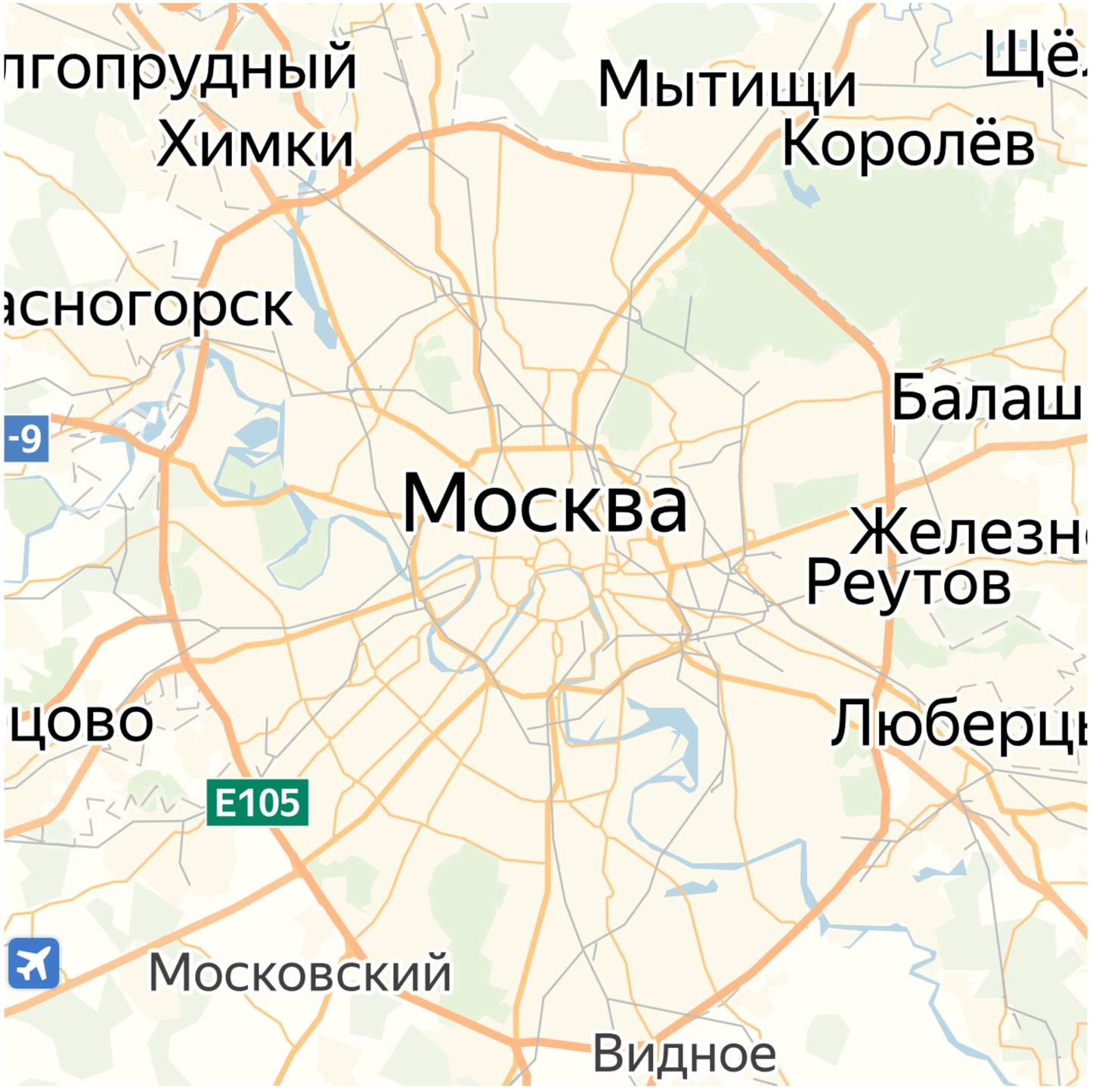
Андрей Роенко, руководитель группы разработки API Яндекс.Карт

План

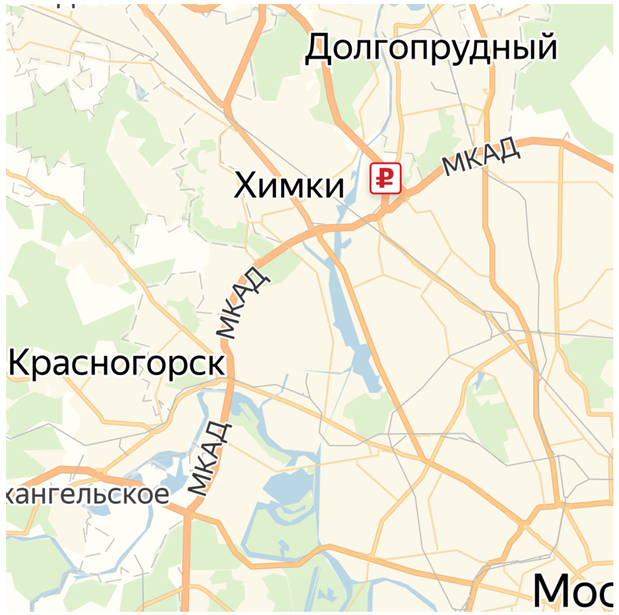
- Опыт Яндекс.Карт
- Что такое WebAssembly?
- Как устроен WebAssembly?
- Как этого хватает?
- Программируем на WebAssembly
- Будущее
- FAQ
- Не только web
- Заключение

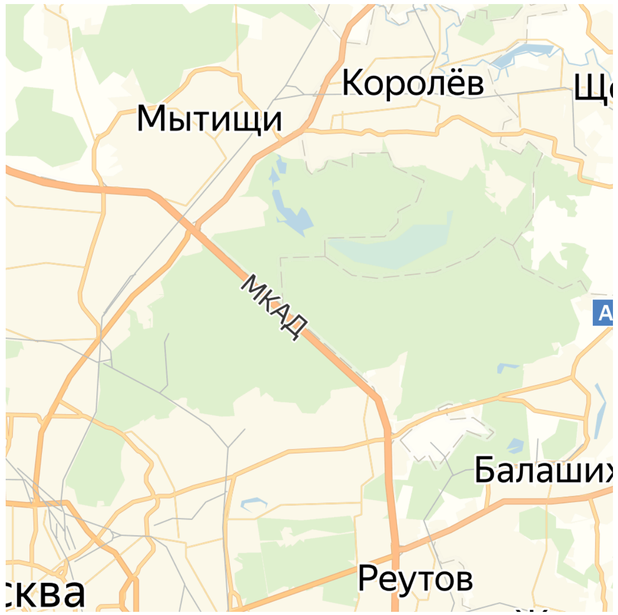
Опыт Яндекс.Карт

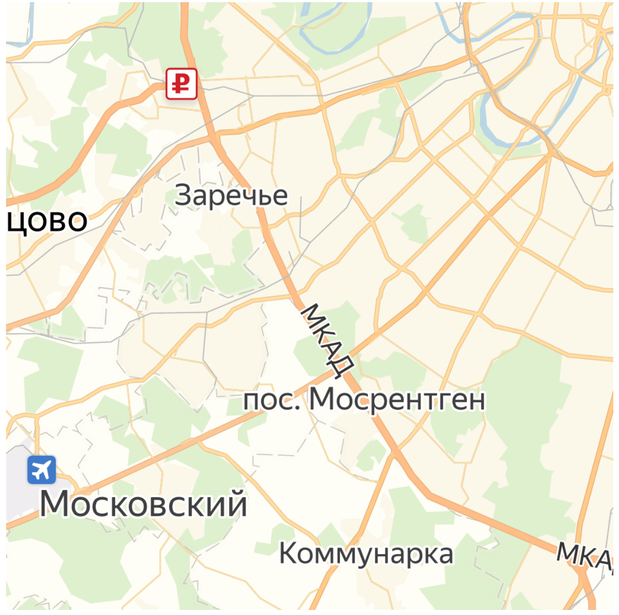

Поиск мест и адресов
Москва
10°C
Доставка еды
Аптеки
Продукты
Где поесть
АЗС
Больницы
Банкоматы
Все места
3,8
Коронавирус
Ивантеевка
Брёхово
Долгопрудный
Королёв
Щёлково
Мытищи
Химки
МКАД
Нахабино
Красногорск
А-103
Балашиха
Архангельское
Москва
Реутов
А-109
Одинцово
Заречье
Люберцы
Власиха
Минское ш.
Малаховка
Дзержинский
пос. Мосрентген
Московский
Киевское ш.
Лыткарино
Коммунарка
Видное
Петровское
Яндекс
© Яндекс
Разместить рекламу
Редактировать карту
Справка
Условия использования
6 км

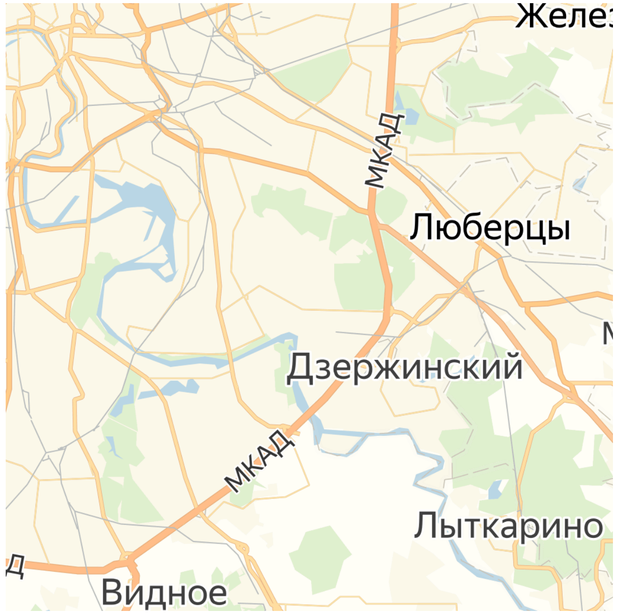

Мытищи
Щё
Химки
Королёв
Балаш
-9
Москва
Железн
Реутов
Люберц
E105
Московский
Видное

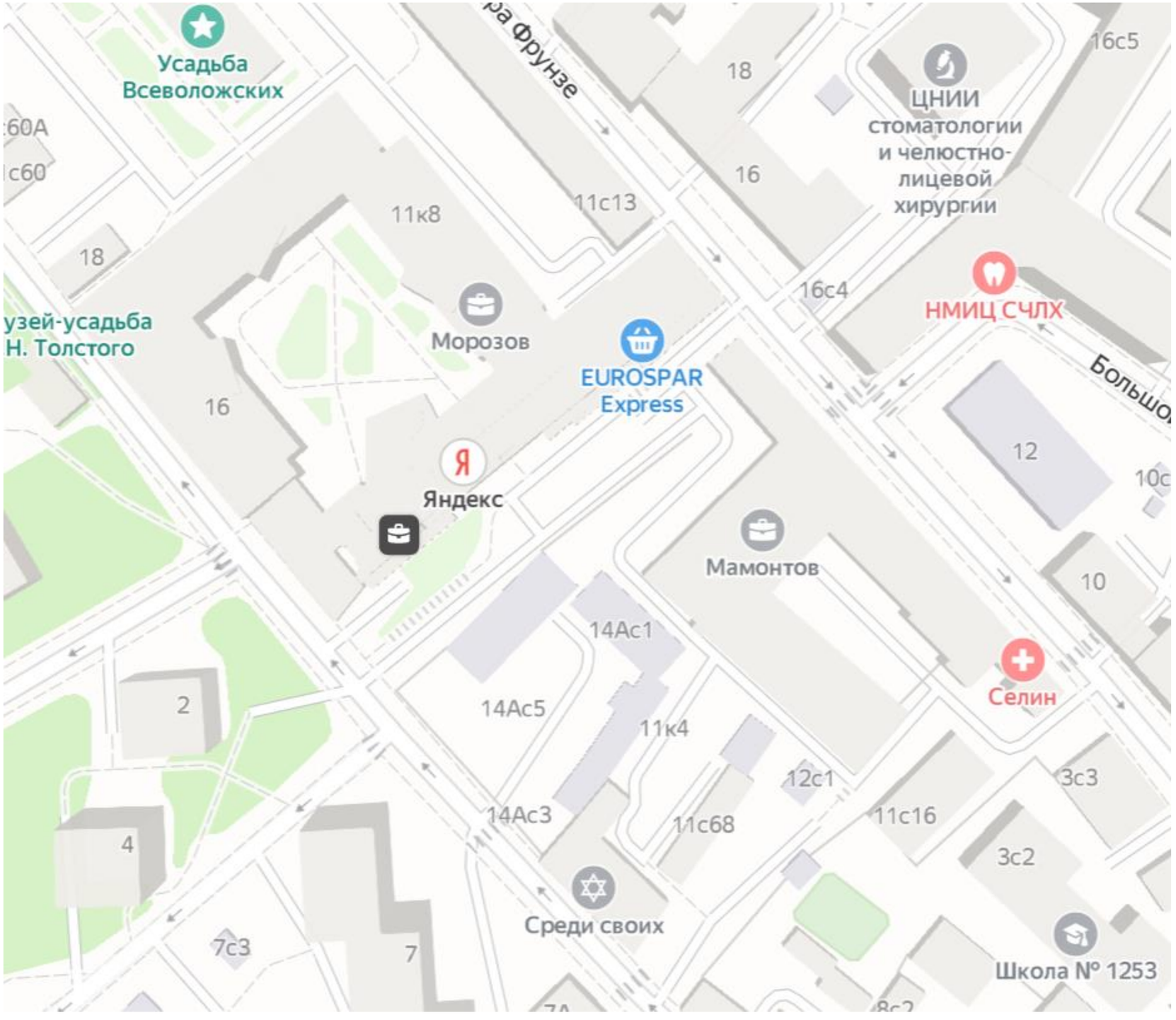

лгопрудный
Химки
Мытищи
Щё
Королёв
асногорск
-9
Балаш
Москва
Железн
Реутов
цово
Люберц
E105
Московский
Видное

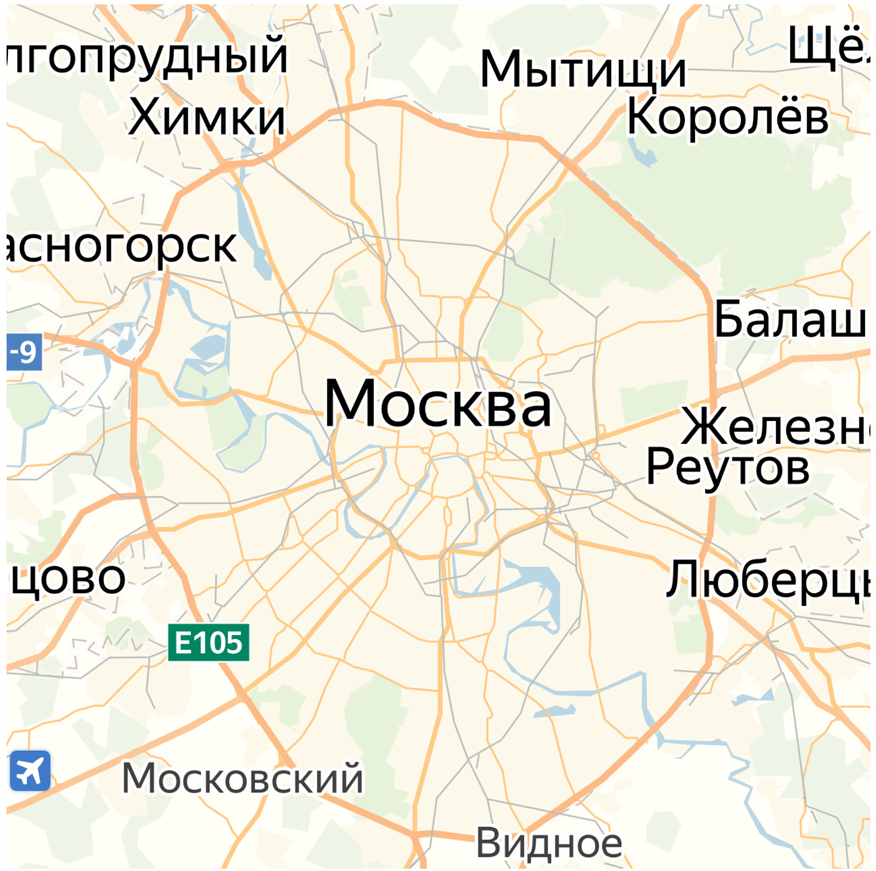

Долгопрудный
Химки
МКАД
МКАД
Красногорск
МКАД
кангельское
Мос


Королёв
Мытищи
МКАД
Балаших
Реутов
сква


Заречье
цово
МКАД
пос. Мосрентген
Московский
Коммунарка
МКА


Желе
МКАД
Люберцы
Дзержинский
МКАД
Лыткарино
Видное


Усадьба Всеволожских
ра Фрунзе
18
16с5
ЦНИИ стоматологии и челюстно-лицевой хирургии
16
11с13
11к8
18
16с4
НМИЦ СЧЛХ
Морозов
EUROSPAR Express
узей-усадьба Н. Толстого
16
Большо
12
10с
Яндекс
Мамонтов
10
14Ас1
Селин
14Ас5
11к4
2
12с1
3с3
14Ас3
11с68
11с16
4
3с2
Среди своих
7с3
7
Школа № 1253

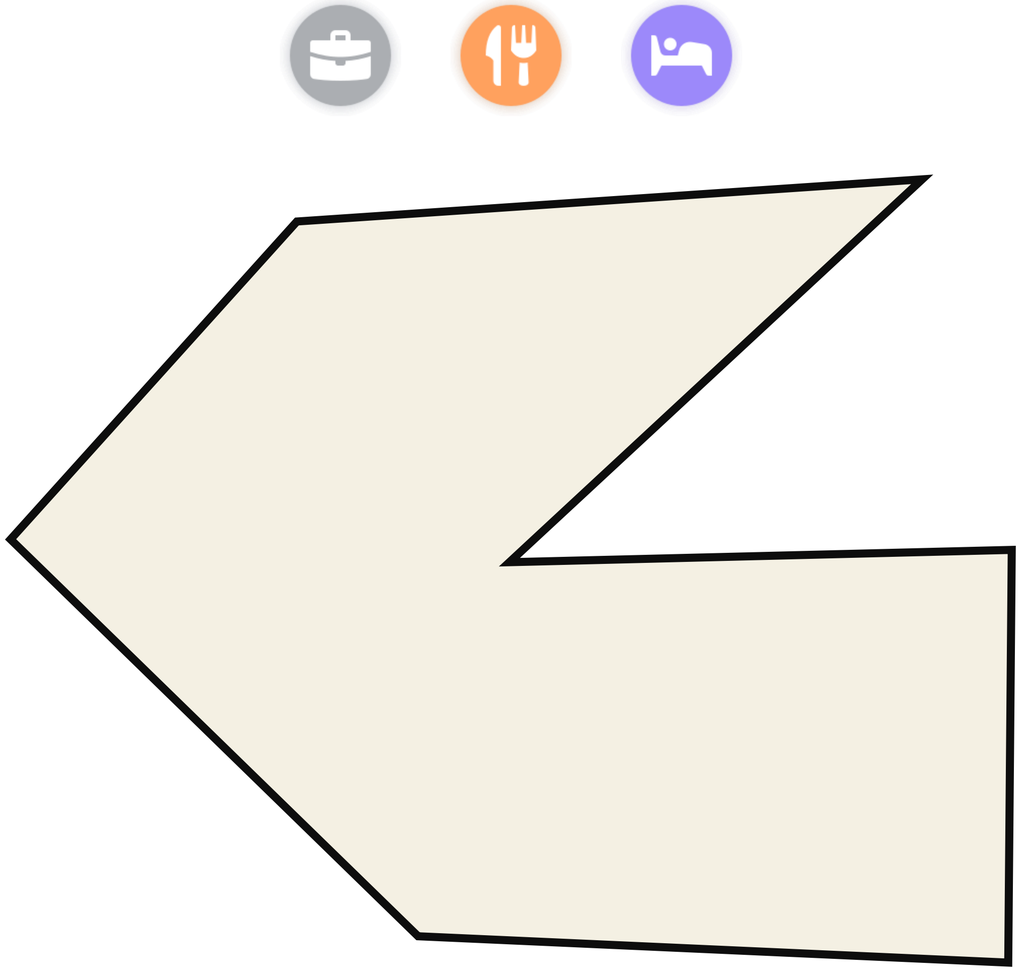


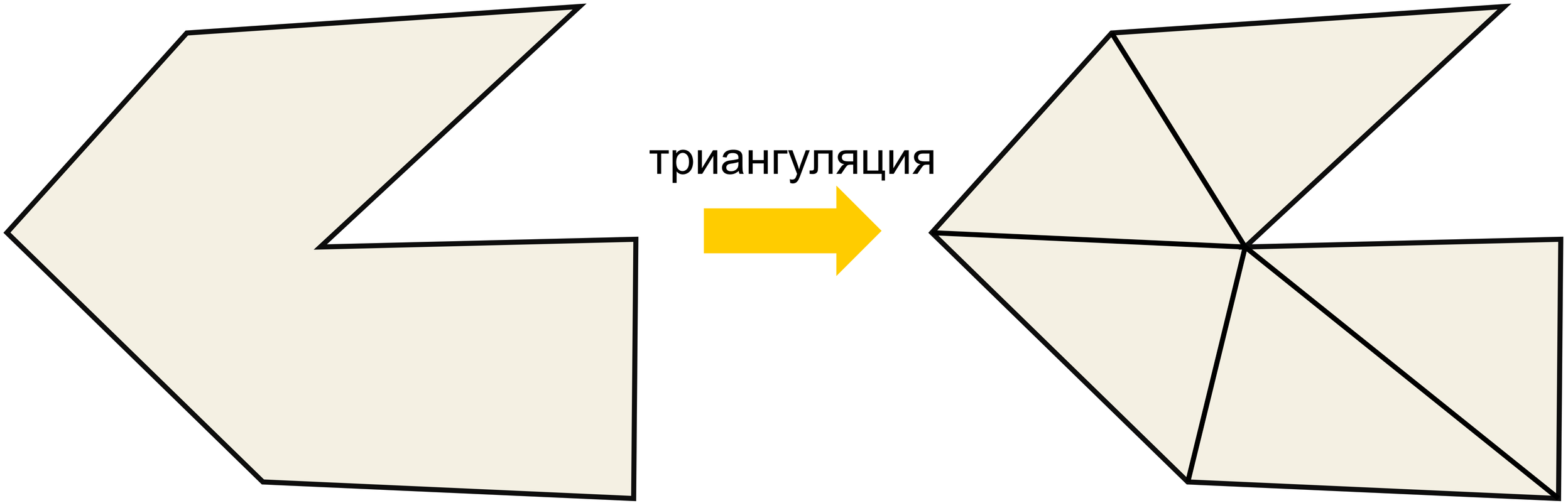
protobuf

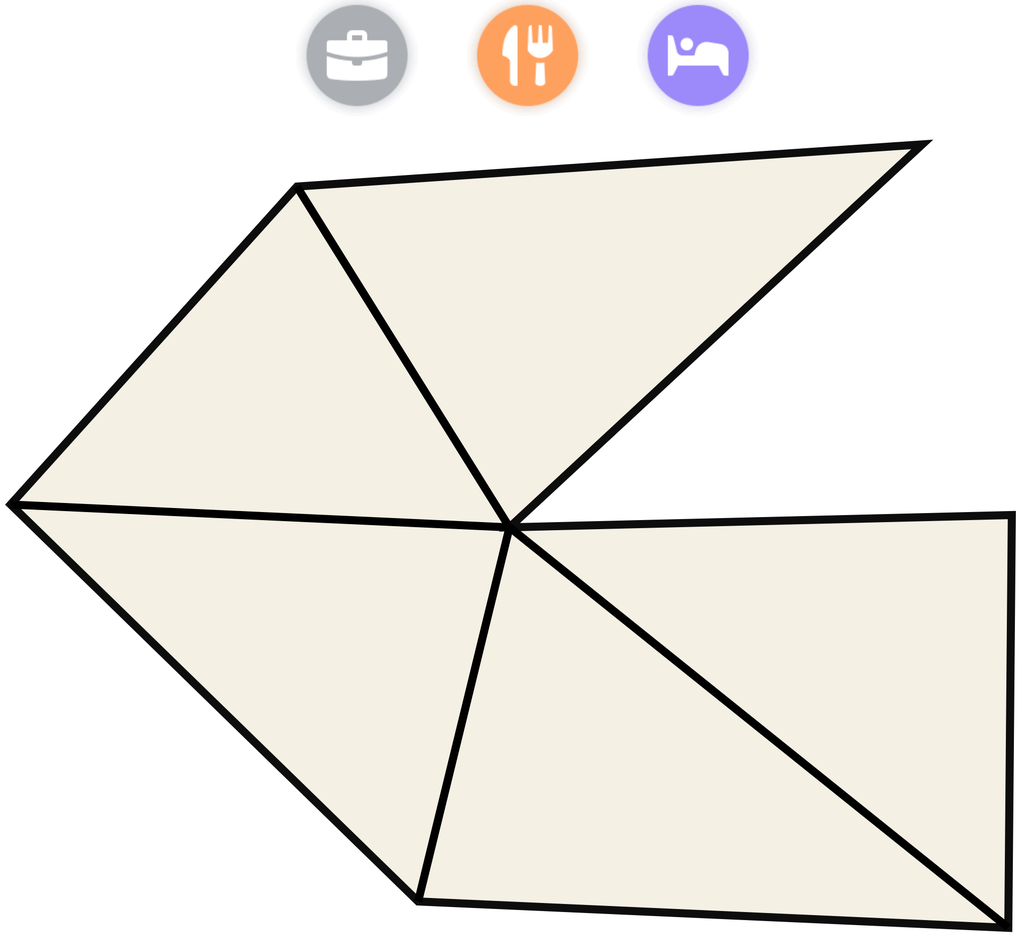
```
1
2  Ä•
3  1••••admin.locality_a.locality_0•••¨••••"•""••ÿðùü
4  >•••(urban-area.landuse_a.urban-residential_0•••Ò••••"•""••ÿòúý
5  2••••vegetation.veg_residential_0•••Ê••••"•""••ÿÎðß
6  +••••vegetation.veg_lawn_0•••ô••••"•""••ÿÎðß
7  8•••"urban-area.landuse_a.sportground_0••••••••"•""••ÿÎðß
8  •••••2label.transport.station.bus.transport_p.bus-stop_0•u•äÃÜ••••"•"*f
9  '
10  GBZT3ZZLMSRGRXRPSVESLHMRHB4E6JBA•••••••••?••••?"'
11  YTPHLN4TZ5NKUWO2XDHVW5JAN6GF2PLE•••••-•••?5•••?2•
12 Æ•••••poi.poi_large_0•°••àÅÜ••••"•"*f
13 '
14  MTOTZ2S6I34O22QPK2HVMBBYBZPSLATZ•••••••••?••••?"'
15  BSLZZQ3SL4EWGBXRGP7TMZ3QJNTB6MDH•••••-•••?5•••?2;
16 9
17 ÿ••••(87931be60dd7d1ced983a8030b11bd153c5e45ed••É•@%ÿÿÿÿ
18 Æ•••••poi.poi_large_0•°••àÅÜ••••"•"*f
19 '
20  YBORFAZQNY6ELZR6UK4AWSR54FUFTJFA•••••••••?••••?"'
21  IB3H6EIXZB4THW26QH7RWURJBNCZWDLC•••••-•••?5•••?2;
22 9
23 ÿ••••(87931be60dd7d1ced983a8030b11bd153c5e45ed••É•@%ÿÿÿÿ
24 Æ•••••poi.poi_large_0•°••àÅÜ••••"•"*f
25 '
```

protobuf

```
1
2  Ä•
3  1••••admin.locality_a.locality_0•••¨••••"•""••ÿðùü
4  >•••(urban-area.landuse_a.urban-residential_0•••Ò••••"•""••ÿòúý
5  2••••vegetation.veg_residential_0•••Ê••••"•""••ÿÎðß
6  +••••vegetation.veg_lawn_0•••ô••••"•""••ÿÎðß
7  8•••"urban-area.landuse_a.sportground_0••••••••"•""••ÿÎðß
8  •••••2label.transport.station.bus.transport_p.bus-stop_0•u•äÃÜ••••"•"*f
9  '
10  GBZT3ZZLMSRGRXRPSVESLHMRHB4E6JBA•••••••••?••••?"'
11  YTPHLN4TZ5NKUWO2XDHVW5JAN6GF2PLE•••••-•••?5•••?2•
12 Æ•••••poi.poi_large_0•°••àÅÜ••••"•"*f
13 '
14  MTOTZ2S6I34O22QPK2HVMBBYBZPSLATZ•••••••••?••••?"'
15  BSLZZQ3SL4EWGBXRGP7TMZ3QJNTB6MDH•••••-•••?5•••?2;
16 9
17 ÿ••••(87931be60dd7d1ced983a8030b11bd153c5e45ed••É•@%ÿÿÿÿ
18 Æ•••••poi.poi_large_0•°••àÅÜ••••"•"*f
19 '
20  YBORFAZQNY6ELZR6UK4AWSR54FUFTJFA•••••••••?••••?"'
21  IB3H6EIXZB4THW26QH7RWURJBNCZWDLC•••••-•••?5•••?2;
22 9
23 ÿ••••(87931be60dd7d1ced983a8030b11bd153c5e45ed••É•@%ÿÿÿÿ
24 Æ•••••poi.poi_large_0•°••àÅÜ••••"•"*f
25 '
```



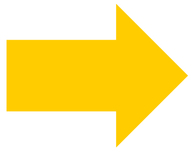

триангуляция



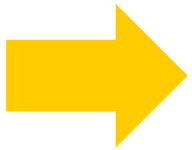

```
88 c6 54 81 b4 4b db 14
81 96 ea 35 66 4c 16 8d
45 65 67 74 65 1e dc 6b
24 d8 a4 6e 59 6b 2d 0e
1a 41 d1 9d b3 7b 91 5f
0f f1 08 f0 5a a7 8d e1
69 57 0a 49 3f 66 a1 8a
35 a8 f4 fb e8 82 db f6
f8 f3 0b 3a 8c 67 0b 5c
b2 71 c6 b1 2d ef 74 60
```

protobuf

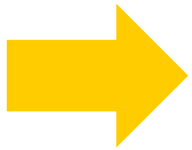

```
bd 47 85 83
13 e3 73 f5
e7 34 f7 55
7a fd 5b 7d
e7 54 6d 40
e3 81 88 cf
52 a3 18 99
fc 05 d7 ea
```



```
6d 39 28 1f
f4 a8 ba 92
88 ea b0 d5
02 2b 90 f0
1a 08 c9 57
18 12 0a d2
0e 42 4e ce
4b eb 79 93
de c8 45 f2
c0 e4 95 a9
```



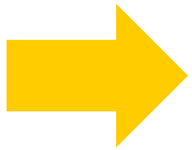
```
b4 c8 fe aa
fd 90 69 73
be 51 57 bf
4a e6 96 77
27 0c 19 77
cc c6 d1 c0
```



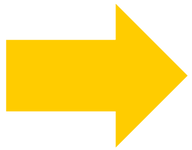
```
22 7e b8 be
6a fb 6a 0c
4d c4 5b 17
ec 7b 4c fb
da 5c 0b 41
44 13 f5 18
a5 69 84 91
93 6a 21 3c
a7 c1 85 f2
de 67 5b d5
69 7b ff d6
42 19 80 e8
```

protobuf

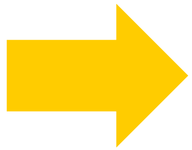

```
bd 47 85 83
13 e3 73 f5
e7 34 f7 55
7a fd 5b 7d
e7 54 6d 40
e3 81 88 cf
52 a3 18 99
fc 05 d7 ea
```



```
6d 39 28 1f
f4 a8 ba 92
88 ea b0 d5
02 2b 90 f0
1a 08 c9 57
18 12 0a d2
0e 42 4e ce
4b eb 79 93
de c8 45 f2
c0 e4 95 a9
```

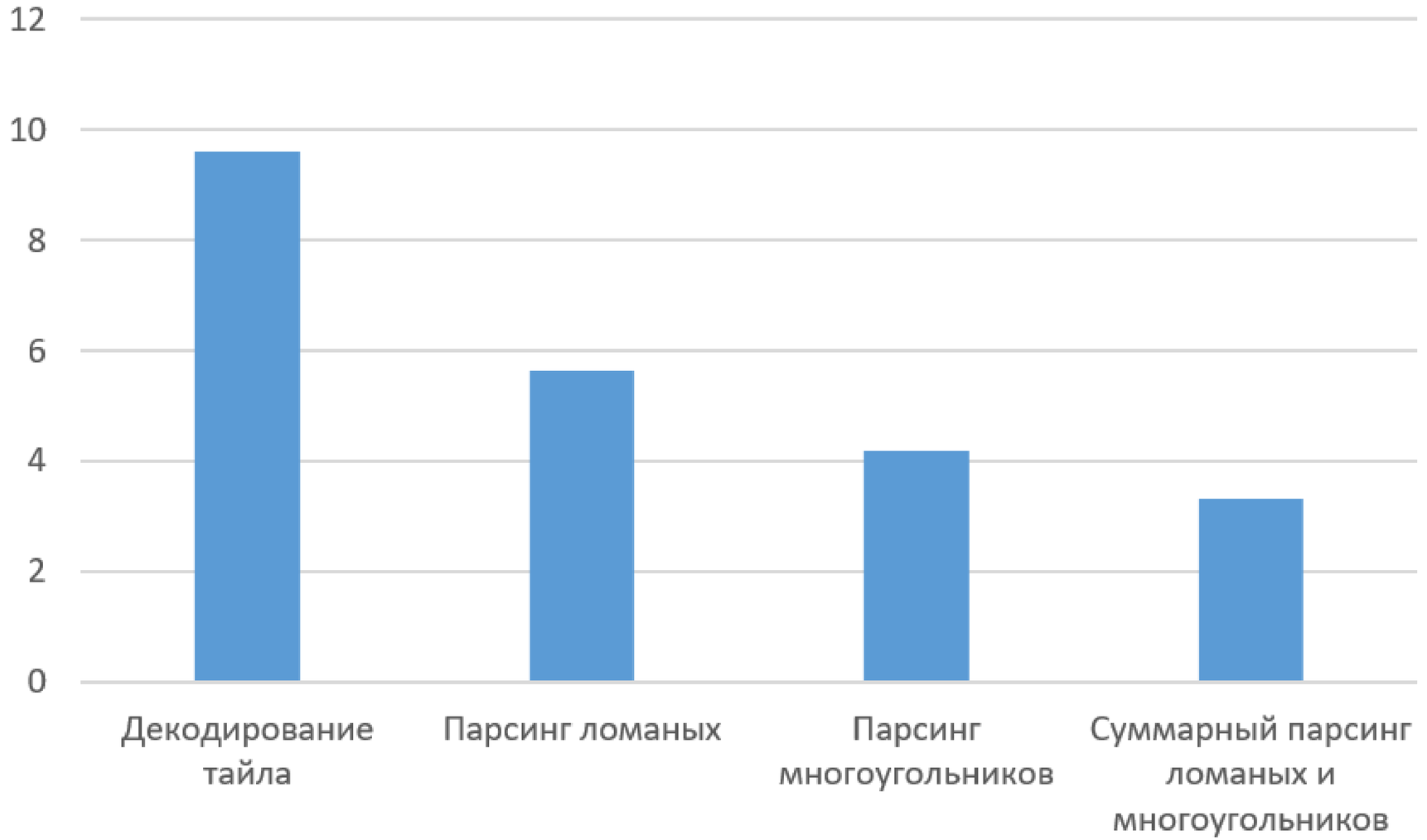


```
b4 c8 fe aa
fd 90 69 73
be 51 57 bf
4a e6 96 77
27 0c 19 77
cc c6 d1 c0
```



```
22 7e b8 be
6a fb 6a 0c
4d c4 5b 17
ec 7b 4c fb
da 5c 0b 41
44 13 f5 18
a5 69 84 91
93 6a 21 3c
a7 c1 85 f2
de 67 5b d5
69 7b ff d6
42 19 80 e8
```

emscripten


12
10
8
6
4
2
0
Декодирование тайла
Парсинг ломаных
Парсинг многоугольников
Суммарный парсинг ломаных и многоугольников

Ускорение отдельных операций – 3х

Ускорение всей работы – 20-25%

Минусы

- Параллельная кодовая база на C++
- Параллельный разработчик на C++
- Параллельные баги на C++
- Сильно увеличивается размер бандла
- Не подходит для всего приложения: много дополнительного кода и накладных расходов

Пока мы так и не решились полностью перевести часть кода на WebAssembly

Подробнее

- Как мы внедряли WebAssembly в Яндекс.Картах и почему оставили JavaScript
- Moscow JS Geo Meetup. Как мы рендерим векторные Яндекс.Карты на вебе

Мы скачали emscripten.
Написали код на C++.
Что-то там скомпилировалось.
На выходе получился какой-то бинарник.
Он как-то запускается в браузере.
Почему-то работает быстро.


WEBASSEMBLY

Что такое WebAssembly?

Что это такое

- Portable, size- and load-time-efficient binary format
- Compilation target
- Can be executed at native speed by taking advantage of common hardware capabilities
- Roughly the same functionality as asm.js
- Primarily aimed at C/C++
- Design to execute within and integrate well with the existing Web platform
- Maintain the versionless, feature-tested and backwards-compatible evolution story of the Web

https://webassembly.org/docs/high-level-goals/


WA

Asm.js

WA

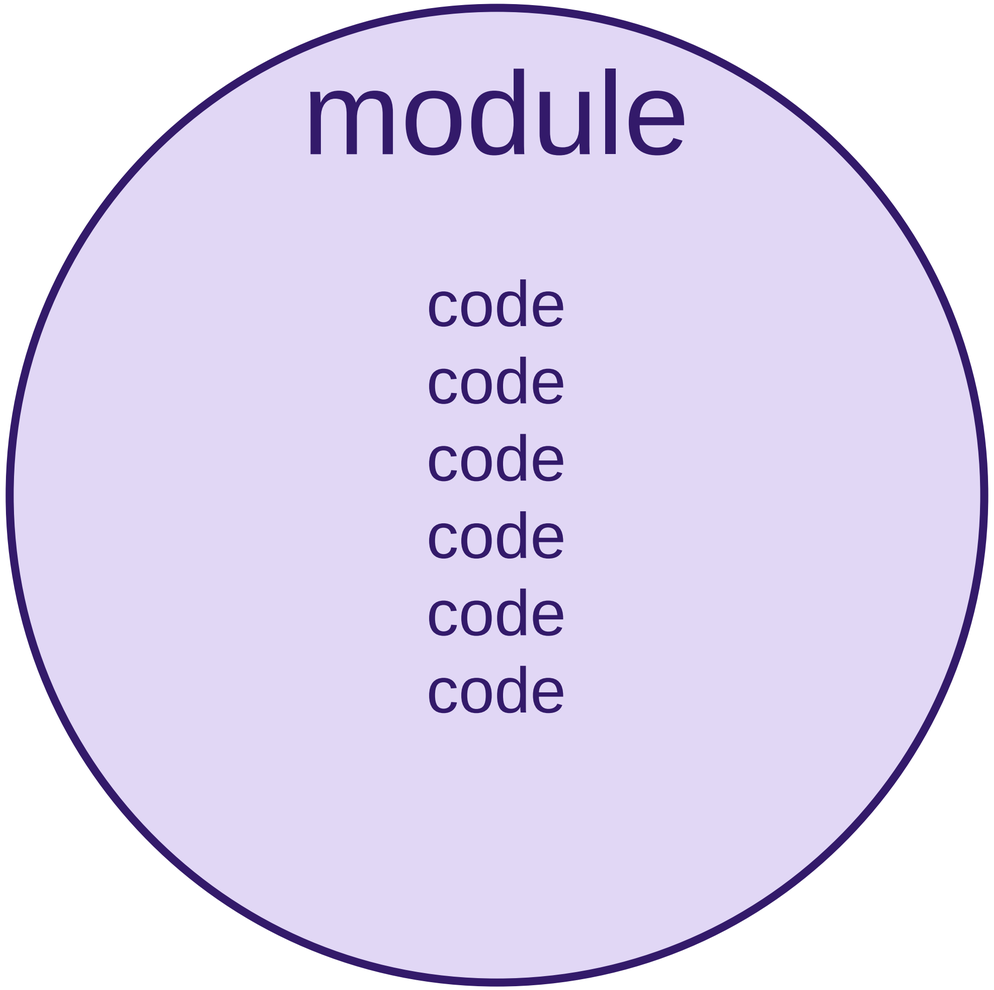
NaCl

Зачем?

	.net jvm flash js	wasm	x86/arm/… nacl
Абстрагирован от конкретного железа?	да	да	нет
Уровень абстракции	высокий	средний	нулевой
Скорость	низкая	средняя	высокая
Система типов	сложная	минималистичная	каша из байтов для команд и данных без обязательной структуры
Структурированность	да	да	
Сложность	сложно	не очень сложно	
Сборка мусора	да	нет	
Дополнительный софт	да	нет	

Как устроен WebAssembly?

Модуль



Типы данных

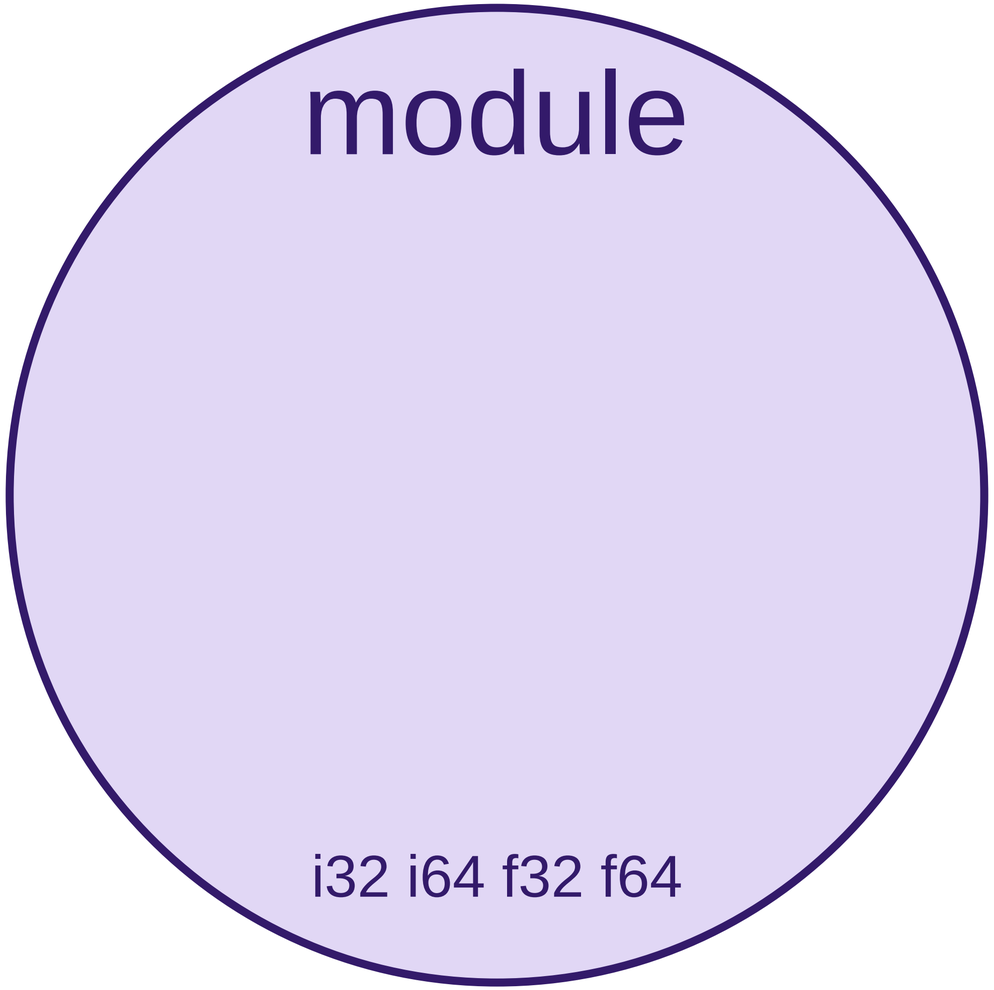
- `f32`, `f64` – числа с плавающей запятой
  - IEEE754
  - По-сути, то же, что `float/double` в C++, `number` в JavaScript

- `i32`, `i64` – целые числа
  - Отрицательные числа представляются через дополнительный код

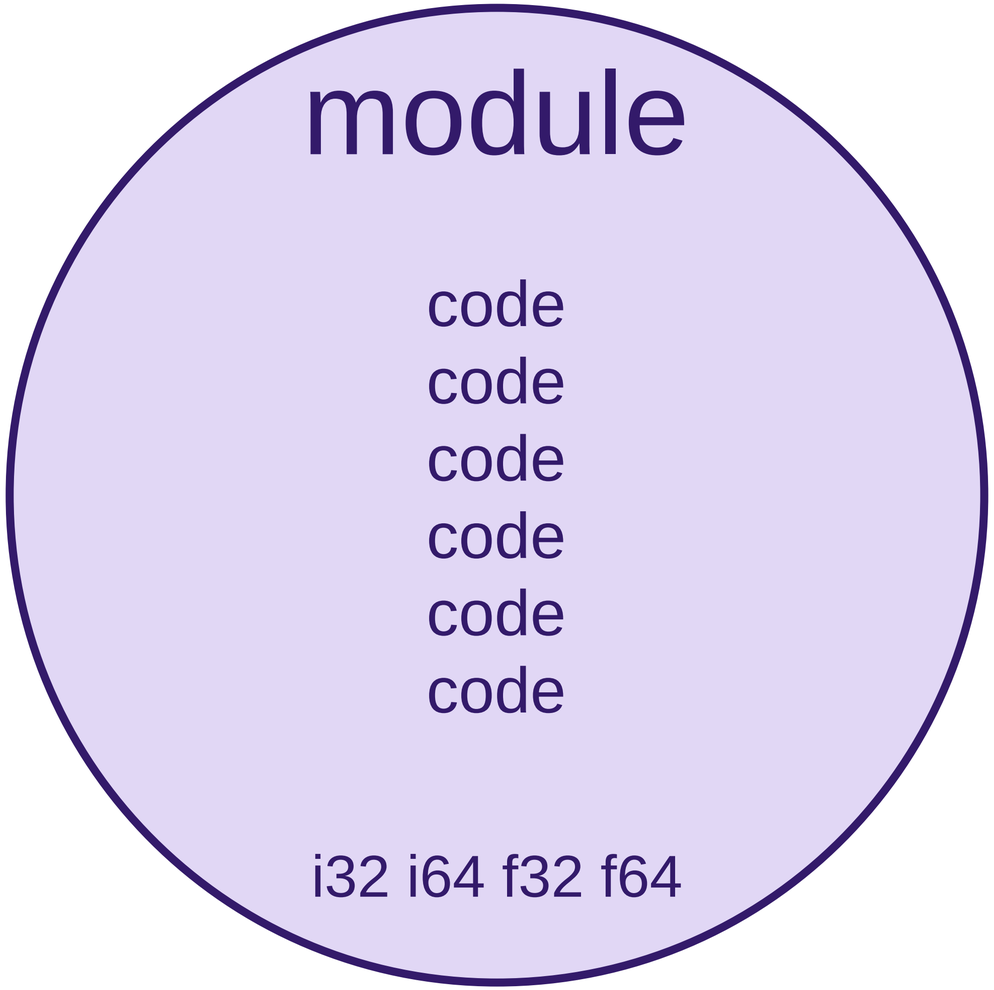
…

- Больше ничего нет
  - Вместо `boolean` – 0 и 1

Типы данных



Инструкции



Инструкции

`type.operation`

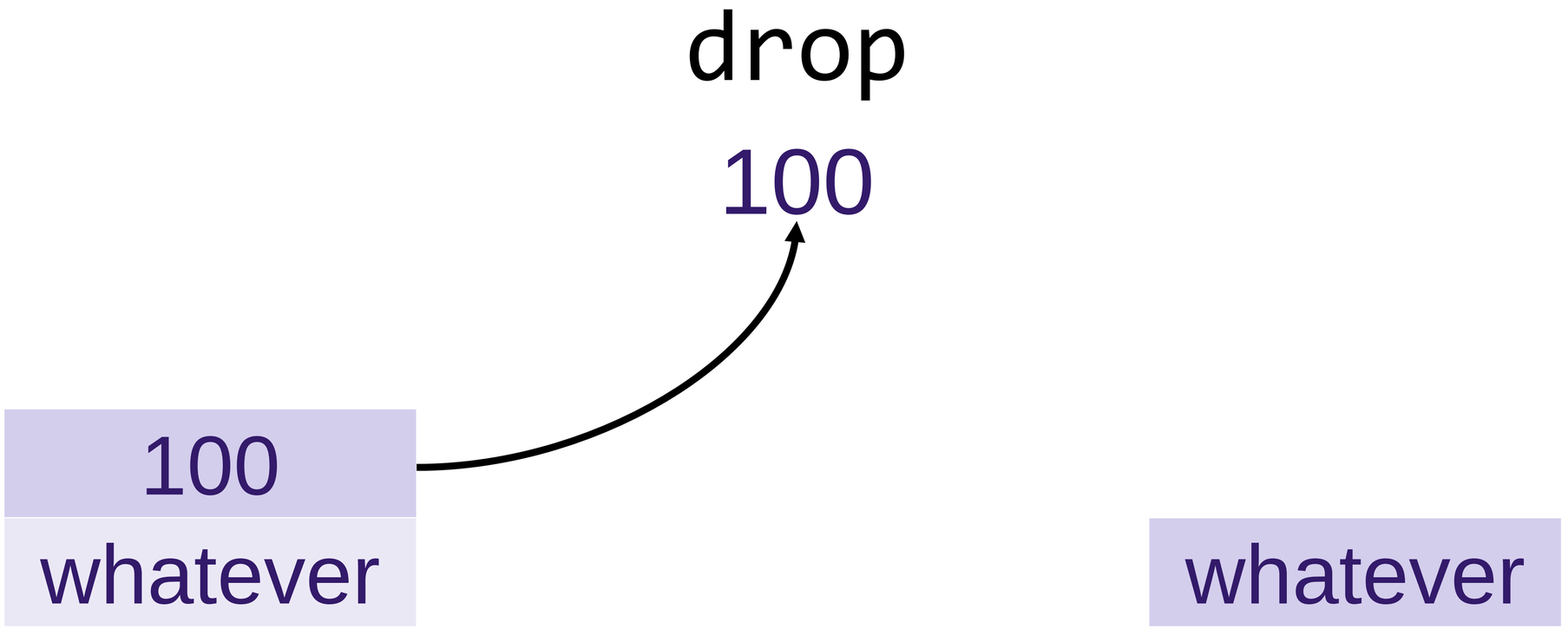
Обычная математика: `add, sub, mul, div, …`
Битовые операции: `and (&), or (|), xor (^), shl/shr (>>/<<), …`
Операции сравнения: `eq (==), eqz (==0), gt/lt (>/<), …`

Стековая машина

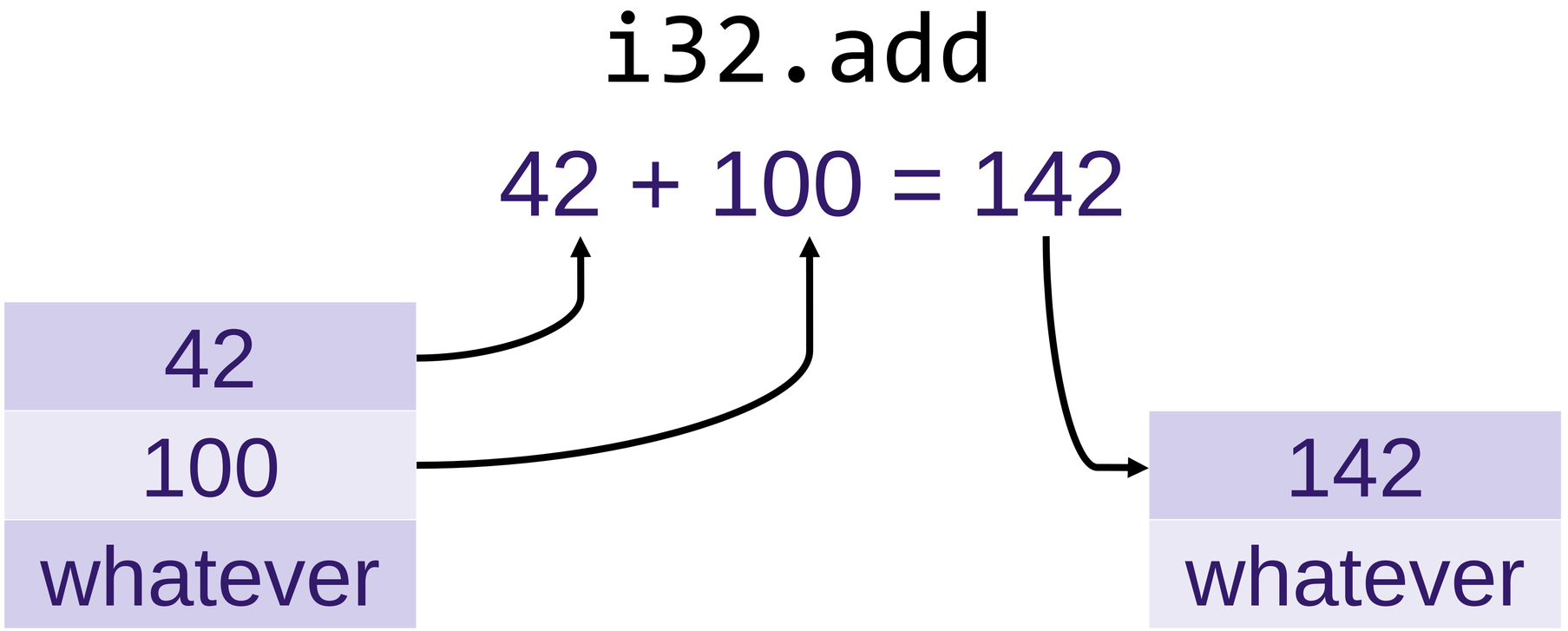
```
i32.const 100
```

100

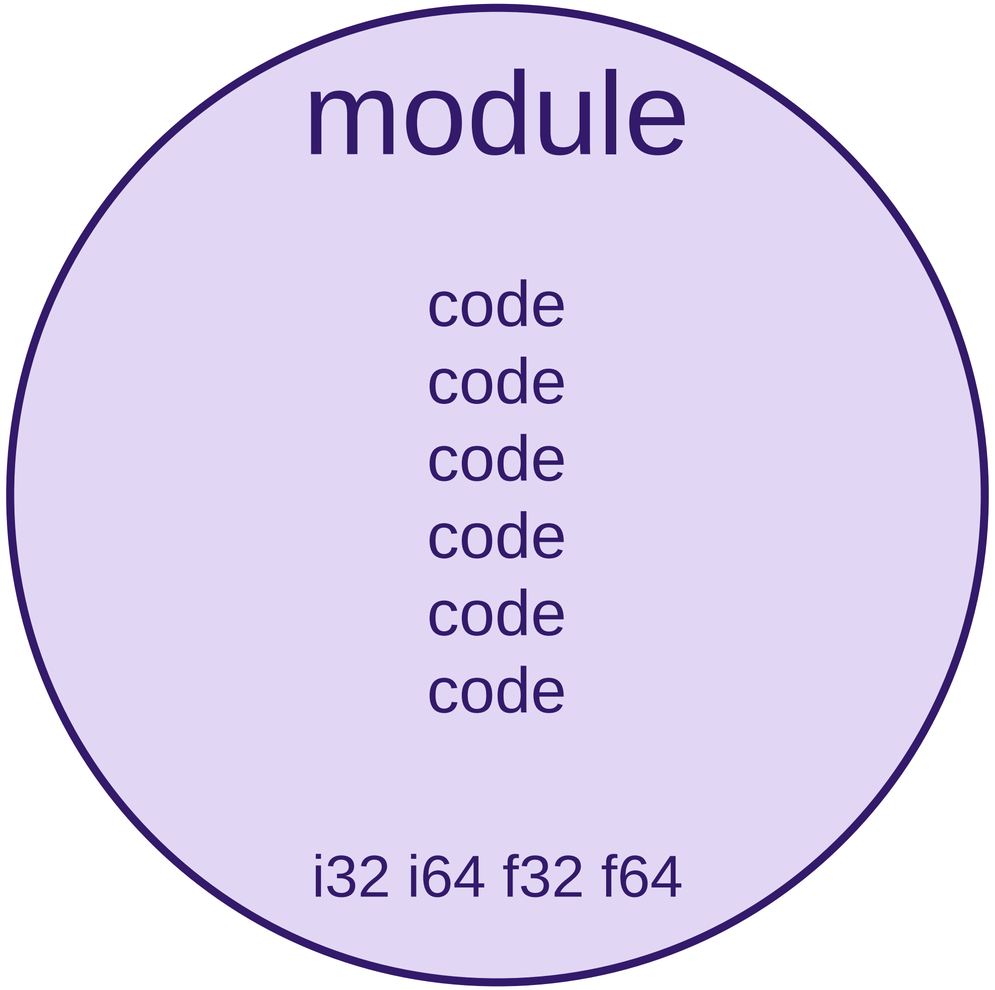
Стековая машина



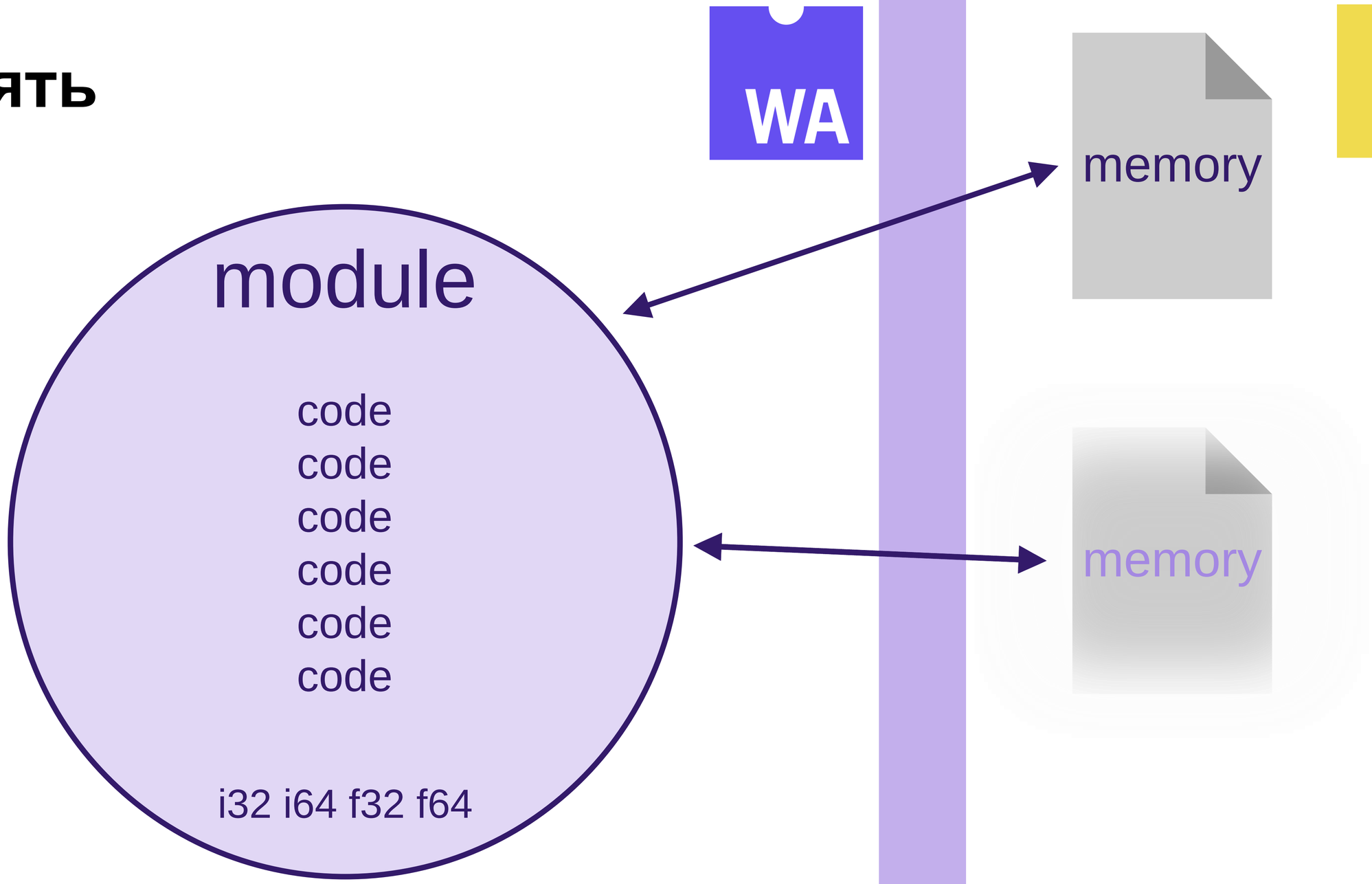
Стековая машина



Интеграция



Память


WA
JS
memory
module
code
code
code
code
code
code
i32 i64 f32 f64
memory

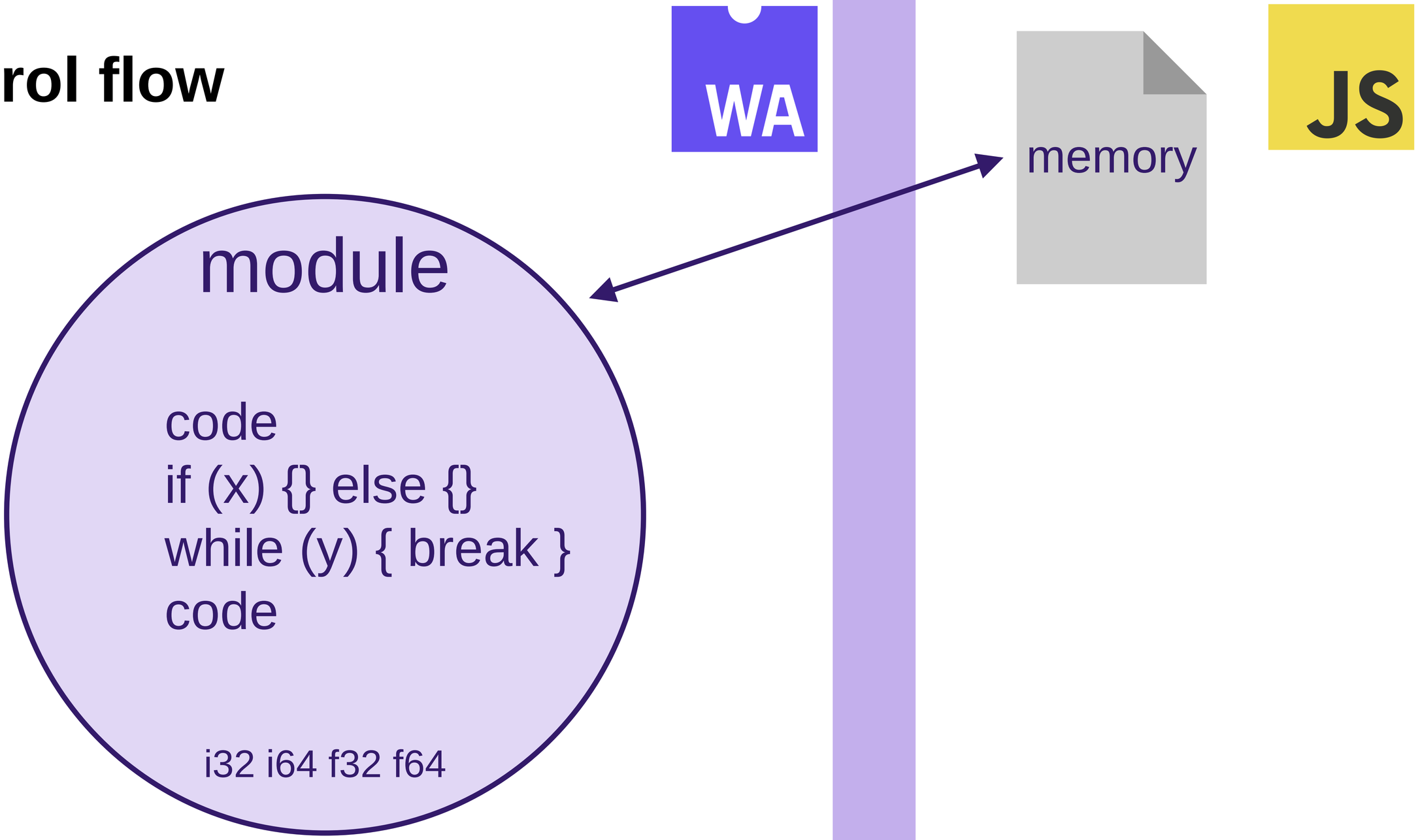
Память

```
auto memory = new char[n * 65536];
```

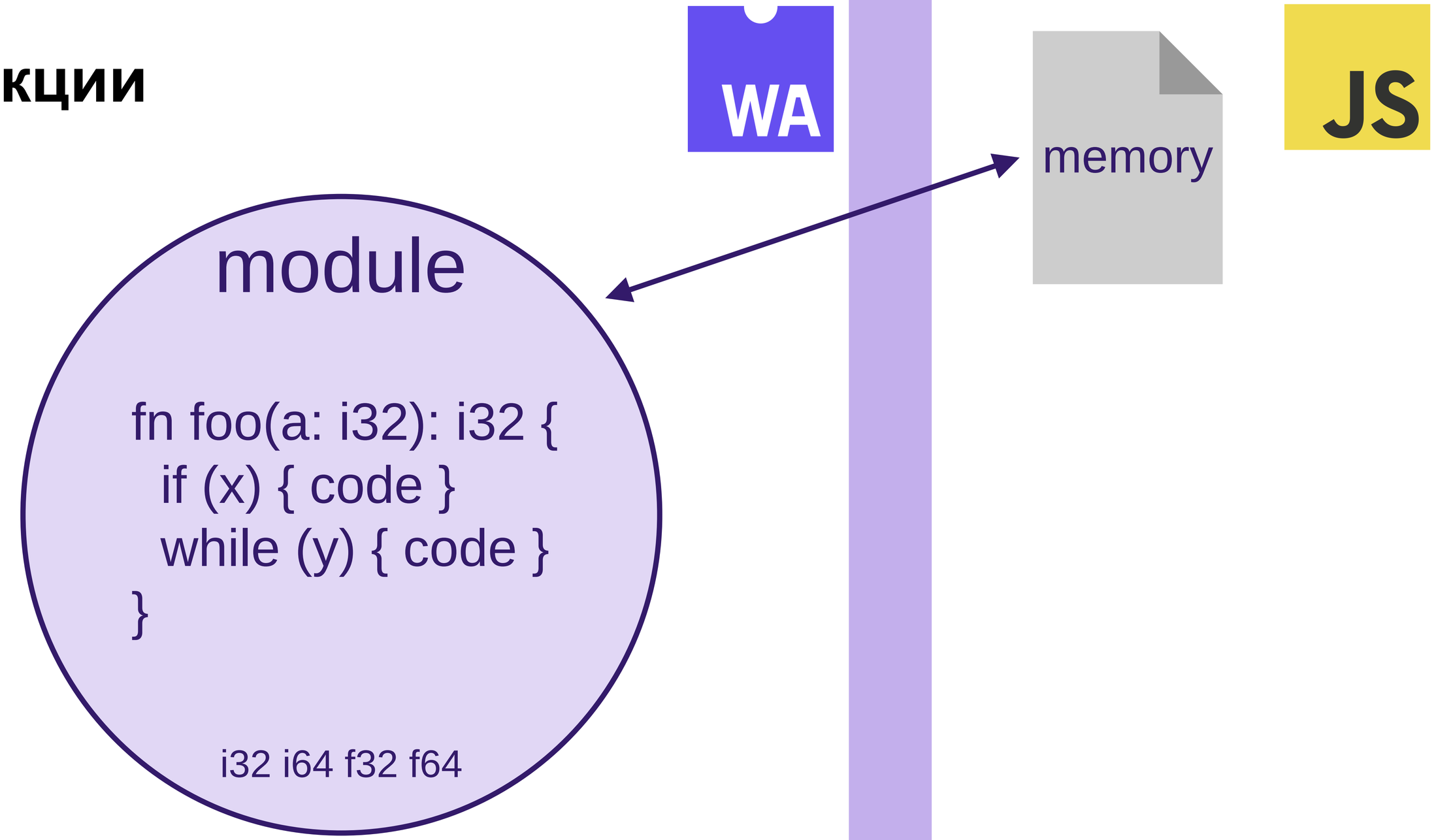
- Можно читать
  - `iXX.load – ((intXX_t*)memory)[index]`
  - `fXX.load – ((float*)memory)[index]`
- Можно писать
  - `iXX.store – ((intXX_t*)memory)[index] = data`
  - `fXX.store – ((float*)memory)[index] = data`

- Память - little endian
  - `0xaabbccdd => dd cc bb aa`

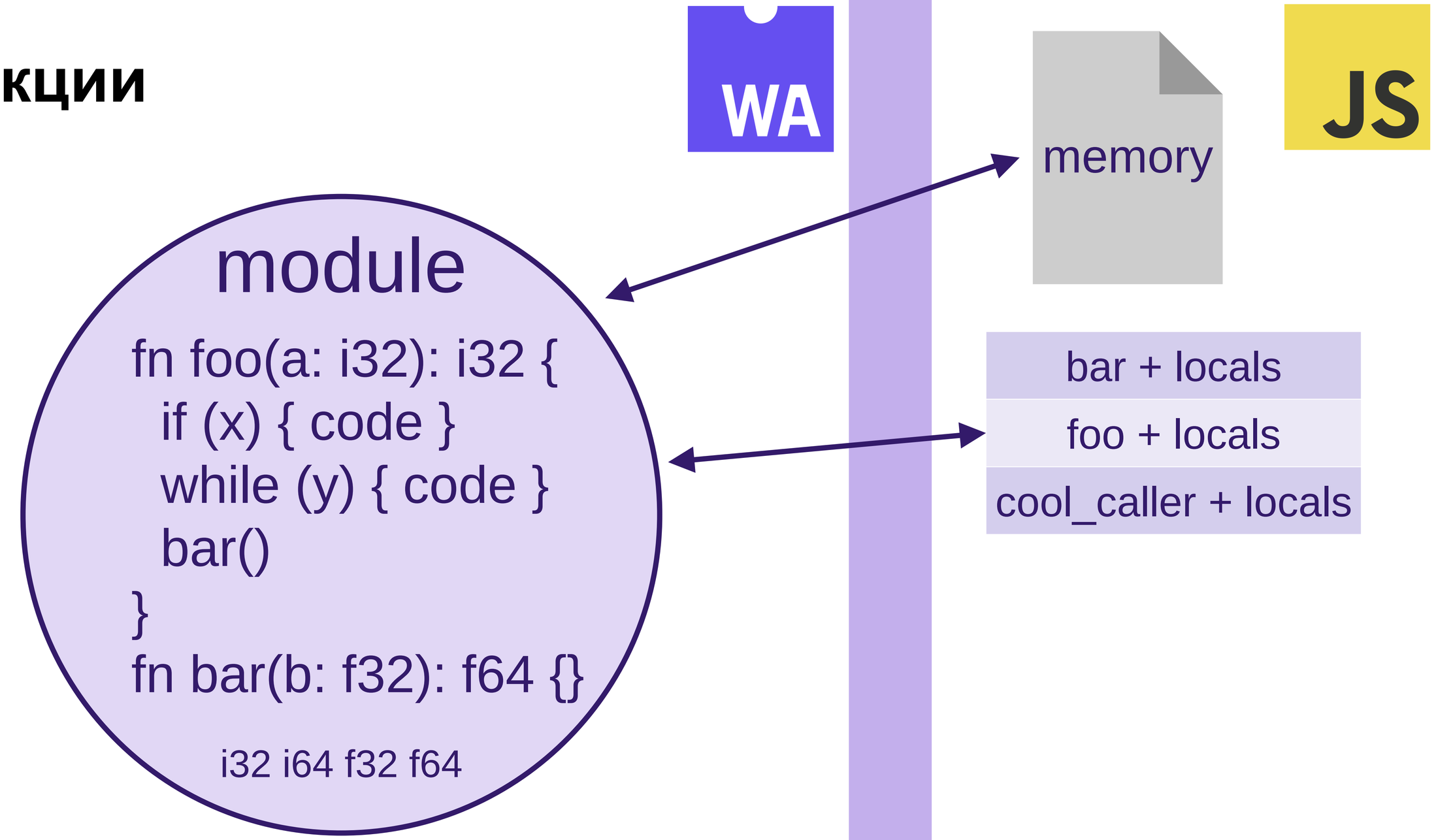
Control flow



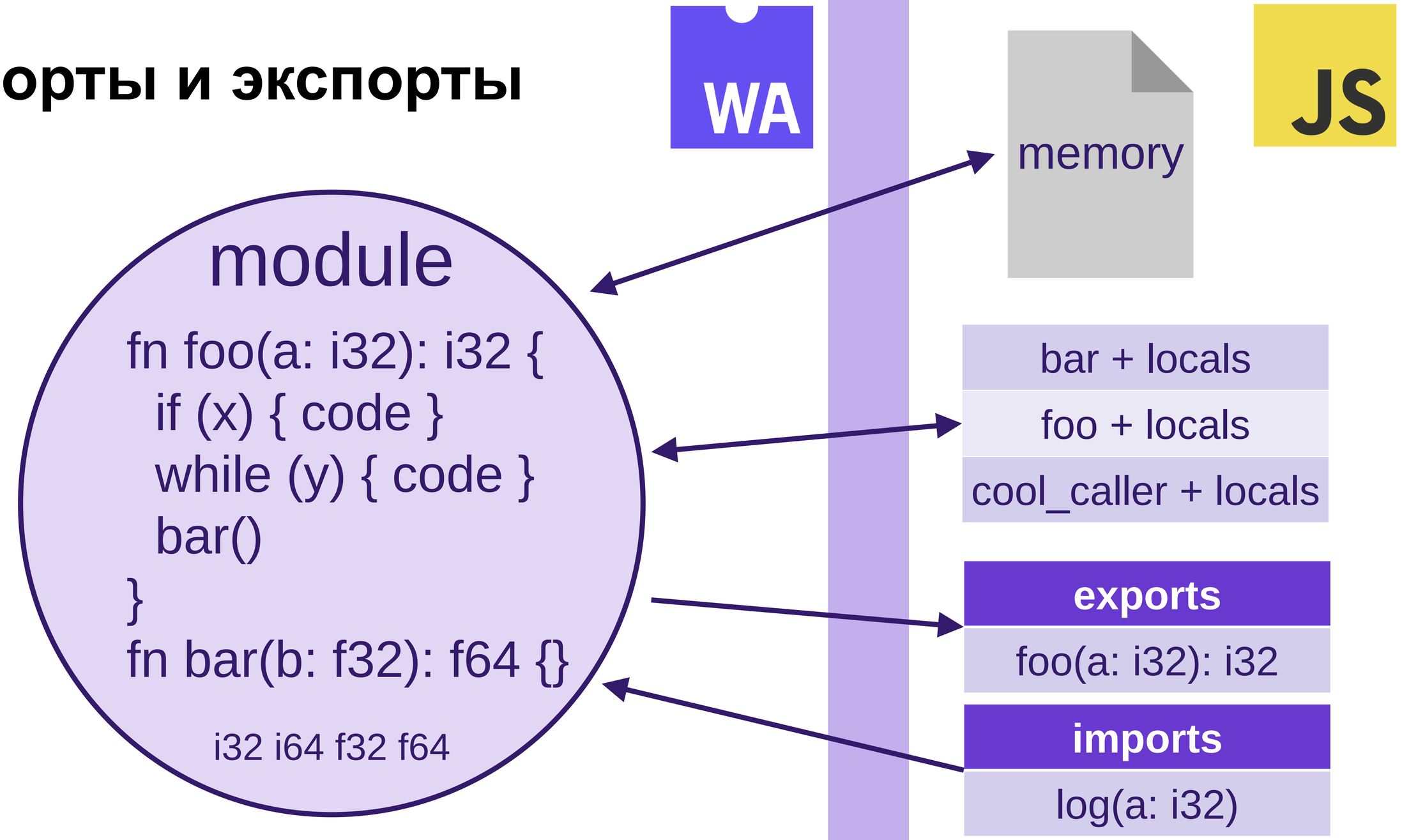
Функции


WA
memory
JS
module
fn foo(a: i32): i32 {
if (x) { code }
while (y) { code }
}
i32 i64 f32 f64

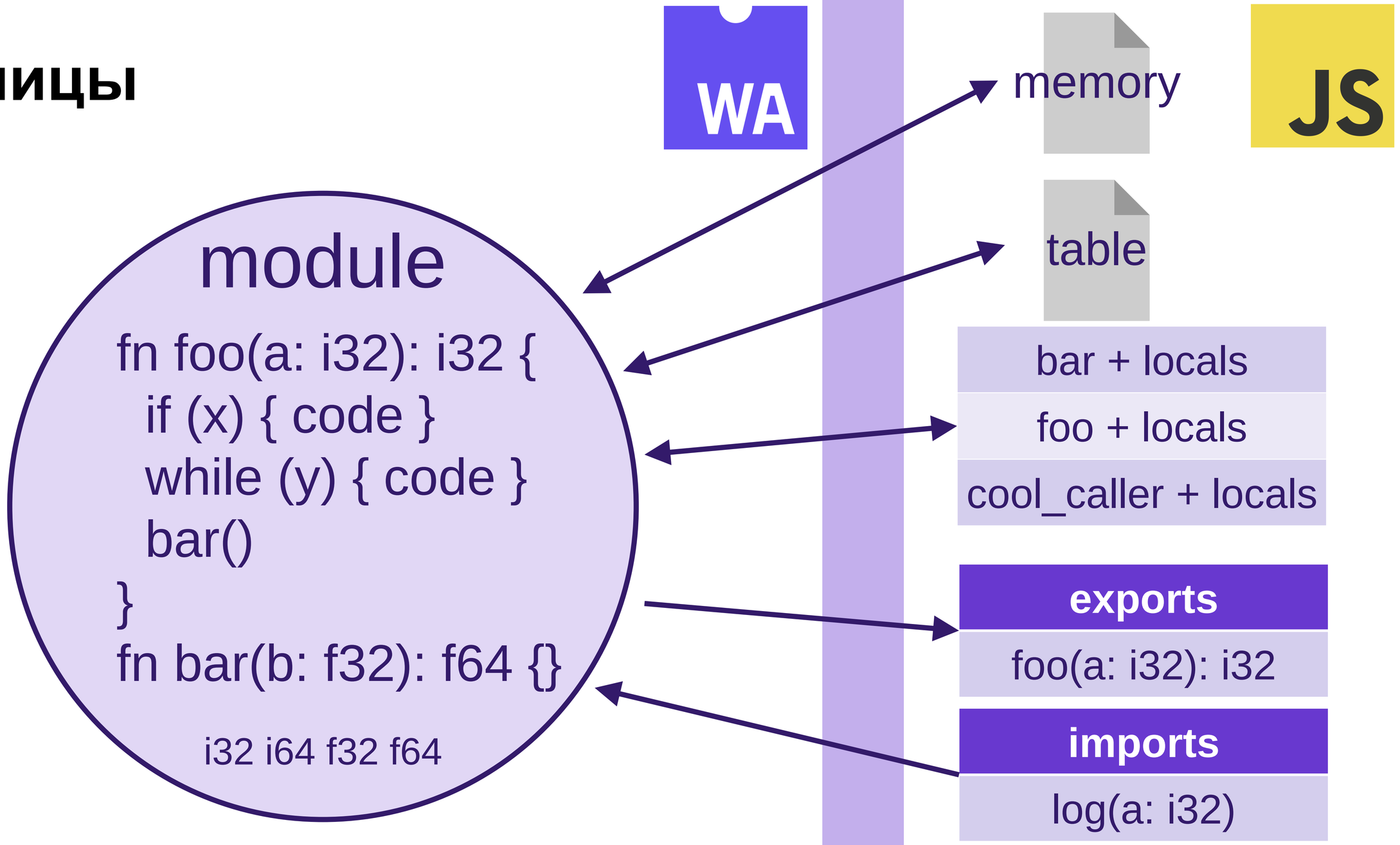
Функции


WA
JS
memory
module
fn foo(a: i32): i32 {
if (x) { code }
while (y) { code }
bar()
}
fn bar(b: f32): f64 {}
i32 i64 f32 f64
bar + locals
foo + locals
cool_caller + locals

Импорты и экспорты



Таблицы


WA
JS
memory
table
module
fn foo(a: i32): i32 {
if (x) { code }
while (y) { code }
bar()
}
fn bar(b: f32): f64 {}
i32 i64 f32 f64
bar + locals
foo + locals
cool_caller + locals
exports
foo(a: i32): i32
imports
log(a: i32)


That's all Folks!
WA

Как этого хватает?

C++ ⇨ wasm

```
import <qt>

export auto createButton(const QString& name) -> QWidget* {
  auto button = new QPushButton(name, this);
  connect(button, &QPushButton::clicked, []() -> void {
    std::cout << "Clicked!\n";
  });

  return button;
}
```

C++ ⇨ wasm

```
import <qt>

export auto createButton(const QString& name) -> QWidget* {
  auto button = new QPushButton(name, this);
  connect(button, &QPushButton::clicked, []() -> void {
    std::cout << "Clicked!\n";
  });

  return button;
}
```

C++ ⇨ wasm

```
import <qt> // QPushButton::QPushButton, connect, QWidget

export auto createButton(const QString& name) -> QWidget* {
  auto button = new QPushButton(name, this);
  connect(button, &QPushButton::clicked, []() -> void {
    std::cout << "Clicked!\n";
  });

  return button;
}
```

C++ ⇨ wasm

```
import <qt> // QPushButton::QPushButton, connect, QWidget

export auto createButton(const QString& name) -> QWidget* {
  auto button = new QPushButton(name, this);
  connect(button, &QPushButton::clicked, []() -> void {
    std::cout << "Clicked!\n";
  });

  return button;
}
```

C++ ⇨ wasm

```
import <qt> // QPushButton::QPushButton, connect, QWidget

export auto createButton(const QString& name) -> QWidget* {
  auto button = new QPushButton(name, this);
  connect(button, &QPushButton::clicked, []() -> void {
    std::cout << "Clicked!\n";
  });

  return button;
}
```

C++ ⇨ wasm

```
import <qt> // QPushButton::QPushButton, connect, QWidget

export auto createButton(const QString& name) -> QWidget* {
  auto button = new QPushButton(name, this);
  connect(button, &QPushButton::clicked, []() -> void {
    std::cout << "Clicked!\n";
  });

  return button;
}
```

C++ ⇨ wasm

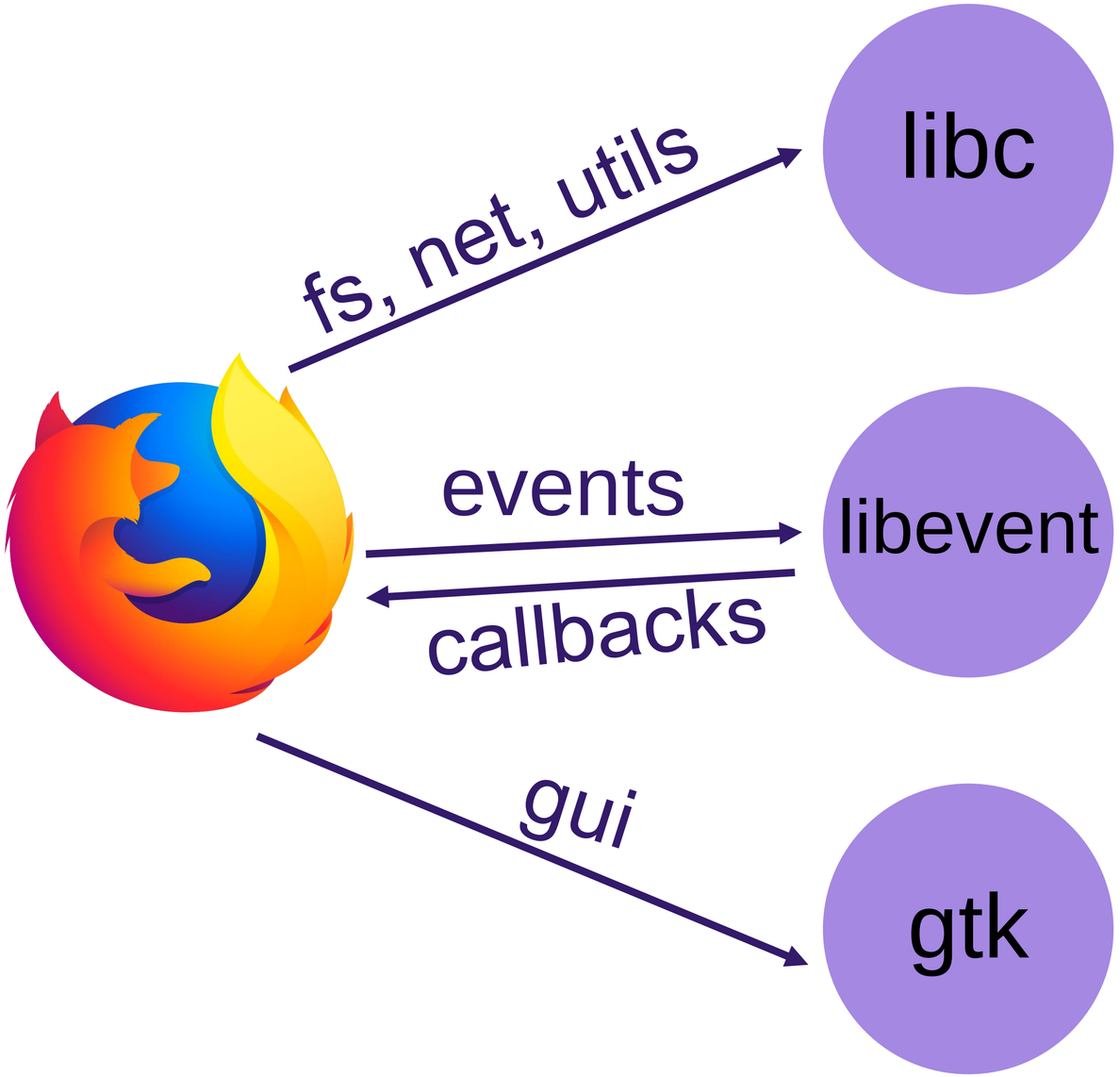
```
import <qt> // QPushButton::QPushButton, connect, QWidget

export auto createButton(const QString& name) -> QWidget* {
  auto button = new QPushButton(name, this);
  connect(button, &QPushButton::clicked, []() -> void {
    std::cout << "Clicked!\n";
  });

  return button;
}
```

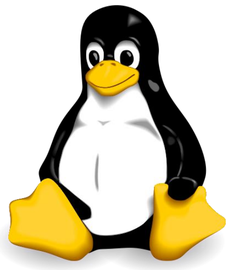
C++ ⇨ wasm

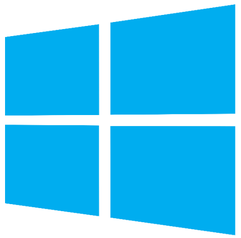
```
import <qt> // QPushButton::QPushButton, connect, QWidget

export auto createButton(const QString& name) -> QWidget* {
  auto button = new QPushButton(name, this);
  connect(button, &QPushButton::clicked, []() -> void {
    std::cout << "Clicked!\n";
  });

  return button;
}
```

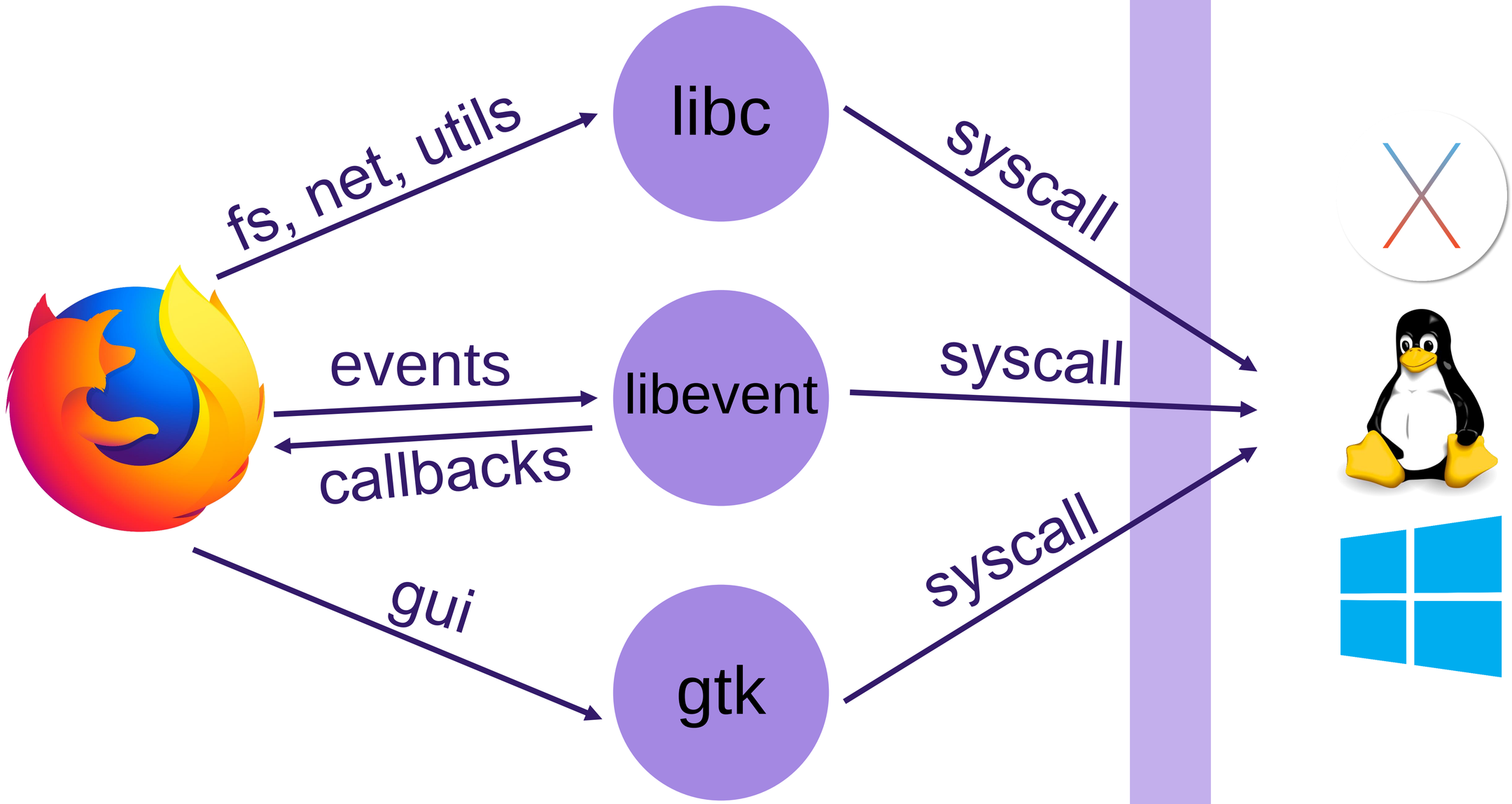
C++ ⇨ wasm

```
import <qt> // QPushButton::QPushButton, connect, QWidget

static auto lambda(bool checked) -> void {
  std::cout << "Clicked!\n";
}

export auto createButton(const QString& name) -> QWidget* {
  auto button = new QPushButton(name, this);
  connect(button, &QPushButton::clicked, lambda);
  return button;
}
```


libc
fs, net, utils
events
libevent
callbacks
gui
gtk






libc
fs, net, utils
syscall
events
libevent
callbacks
syscall
gui
gtk
syscall

Программируем на WebAssembly

Вы скорее всего не будете писать WebAssembly руками

Простой модуль

```
(module
  (export "foo" (func $foo))
  (func $foo (result i32)
    i32.const 42
  )
)
```

Интеграция с JS

```
const src = 'src.wat';
const data = fs.readFileSync(src, 'utf8');

// Compile wat -> wasm
const wasm = wabt.parseWat(src, data);
const {buffer} = wasm.toBinary({});

// Run
const {instance} = await WebAssembly.instantiate(buffer);
const e = instance.exports;
console.log(e.foo()); // 42
```

Параметры функций

```
(module
  (export "foo" (func $foo))
  (func $foo (param i32) (param i32) (result i32)
    ;;         ^^ 0         ^^ 1
    ;; https://webassembly.github.io/spec/core/exec/instructions.html#exec-local-get
    local.get 0 ;; []          => [lhs]
    local.get 1 ;; [lhs]       => [lhs, rhs]

    ;; https://webassembly.github.io/spec/core/exec/instructions.html#exec-binop
    i32.add     ;; [lhs, rhs] => [lhs + rhs]
  )
)



console.log(e.foo(1, 2)); // 3
```

Именованные параметры

```
(module
  (export "foo" (func $foo))
  (func $foo (param $lhs i32) (param $rhs i32) (result i32)
    ;;               ^^^^ 0           ^^^^ 1
    ;; https://webassembly.github.io/spec/core/exec/instructions.html#exec-local-get
    ;; foo(1, 2)
    local.get $lhs
    local.get $rhs

    ;; https://webassembly.github.io/spec/core/exec/instructions.html#exec-binop
    i32.add
  )
)

console.log(e.foo(1, 2)); // 3
```

Доллары – сахарок

```
(module
  (export "foo" (func 0))
  (func (param i32) (param i32) (result i32)
    ;;   ^^ 0         ^^ 1
    ;; https://webassembly.github.io/spec/core/exec/instructions.html#exec-local-get
    ;; foo(1, 2)
    local.get 0
    local.get 1

    ;; https://webassembly.github.io/spec/core/exec/instructions.html#exec-binop
    i32.add
  )
)

console.log(e.foo(1, 2)); // 3
```

```
(module
  (export "foo" (func $foo))
  (func $foo (param $lhs i32) (param $rhs i32) (result i32)
    ;;              ^^^^ 0           ^^^^ 1
    ;; https://webassembly.github.io/spec/core/exec/instructions.html#exec-local-get
    ;; foo(1, 2)
    local.get $lhs
    local.get $rhs

    ;; https://webassembly.github.io/spec/core/exec/instructions.html#exec-binop
    i32.add
  )
)

console.log(e.foo(1, 2)); // 3
```

S-Expressions

```
(module
  (export "foo" (func $foo))
  (func $foo (param $lhs i32) (param $rhs i32) (result i32)
    ;; https://webassembly.github.io/spec/core/text/instructions.html#folded-instructions
    (i32.add              ;; 3. [lhs, rhs] => [lhs + rhs]
      (local.get $lhs) ;; 1. []          => [lhs]
      (local.get $rhs) ;; 2. [lhs]       => [lhs, rhs]
    )
  )
)
```

```
console.log(e.foo(1, 2)); // 3
```

S-Expressions – сахарок

```
(module
  (export "foo" (func $foo))
  (func $foo (param $lhs i32) (param $rhs i32) (result i32)
    ;; https://webassembly.github.io/spec/core/text/instructions.html#folded-instructions
    (AAAAA
      BBBBB
      CCCCC
    )
    ;; ==
    BBBBB
    CCCCC
    AAAAA
  )
)
```

S-Expressions

```
(module
  (export "foo" (func $foo))
  (func $foo (param $lhs i32) (param $rhs i32) (result i32)
    ;; https://webassembly.github.io/spec/core/text/instructions.html#folded-instructions
    (i32.add
      (i32.add (local.get $rhs) (local.get $lhs))
      (i32.const 100)
    )
  )
)
```

```
console.log(e.foo(1, 2)); // 103
```

Локальные переменные

```
(module
  (export "foo" (func $foo))
  (func $foo (param $lhs i32) (param $rhs i32) (result i32)
    ;;                ^^^^ 0              ^^^^ 1
    (local $x i32)
    ;;      ^^ 2
    (local.set $x
      (i32.add
        (i32.add (local.get $rhs) (local.get $lhs))
        (i32.const 100)))
    ;; ...
    local.get $x
  )
)
console.log(e.foo(1, 2)); // 103
```

Ветвление

```
(module
  (export "foo" (func $foo))
  (func $foo (param $cond i32) (result i32)
    (local $x i32)

    ;; https://webassembly.github.io/spec/core/text/instructions.html#folded-instructions
    (if (i32.gt_s (local.get $cond) (i32.const 50))
      (then (local.set $x (i32.const 1)))
      (else (local.set $x (i32.const 0)))
    )

    local.get $x
  )
)
console.log(e.foo(50)); // 1
console.log(e.foo(-10)); // 0
```

Ветвление

```
(module
  (export "foo" (func $foo))
  (func $foo (param $cond i32) (result i32)
    (local $x i32)
    (i32.gt_s (local.get $cond) (i32.const 50)) ;; []    => [0/1]
    if                                          ;; [0/1] => []
    (local.set $x (i32.const 1))
    else
    (local.set $x (i32.const 0))
    end

    local.get $x
  )
)
console.log(e.foo(50)); // 1
console.log(e.foo(-10)); // 0
```

Ветвление

```
(module
  (export "foo" (func $foo))
  (func $foo (param $cond i32) (result i32)
    (local $x i32)
    (i32.gt_s (local.get $cond) (i32.const 50)) ;; []    => [0/1]
    if                                          ;; [0/1] => []
    (local.set $x (i32.const 1))
    else                                        ;; x86
    (local.set $x (i32.const 0))                ;;   cmp
    end                                         ;;   jxx else
                                                ;; then:
    local.get $x                                ;;   ...
  )                                             ;;   jmp end
)                                               ;; else:
console.log(e.foo(50)); // 1                    ;;   ...
console.log(e.foo(-10)); // 0                   ;; end:
                                                ;;   ...
```

Циклы

```
(module
  (export "fib" (func $fib))
  (func $fib (param $n i32) (result i32)
    ;; ...
    ;; https://webassembly.github.io/spec/core/exec/instructions.html#exec-block
    (block (loop
    ;; 1    0
      (br_if 1 (i32.eqz (local.get $n)))

      local.get $curr
      (local.set $curr (i32.add (local.get $prev) (local.get $curr)))

      local.set $prev
      (local.set $n (i32.sub (local.get $n) (i32.const 1)))
      (br 0)
    ))
)
console.log(e.fib(10)); // 55
```

Циклы

```
(module
  (export "fib" (func $fib))
  (func $fib (param $n i32) (result i32)
    ;; ...
    ;; https://webassembly.github.io/spec/core/exec/instructions.html#exec-block
    (block (loop
    ;; 1    0
      (br_if 1 (i32.eqz (local.get $n)))

      local.get $curr
      (local.set $curr (i32.add (local.get $prev) (local.get $curr)))

      local.set $prev
      (local.set $n (i32.sub (local.get $n) (i32.const 1)))
      (br 0)
    ))
)
console.log(e.fib(10)); // 55
```

```
;; x86
;; start_of_block:
;;   jmp start_of_block ;; continue
;;   jmp end_of_block   ;; break
;; end_of_block:
```

Вызов функций

```
(module
  (export "fib" (func $fib))
  (func $fib (param $n i32) (result i32)
    (if (i32.eq (local.get $n) (i32.const 0)) (then (return (i32.const 0))))
    (if (i32.eq (local.get $n) (i32.const 1)) (then (return (i32.const 1))))

    (i32.add
      (call $fib
        (i32.sub (local.get $n) (i32.const 2)))
      (call $fib
        (i32.sub (local.get $n) (i32.const 1)))
    )
  )
)
console.log(e.fib(10)); // 55
```

Вызов импортированных функций

```
(module
  (import "env" "log_i32" (func $log_i32 (param i32)))
  (export "foo" (func $foo))

  (func $foo (param $n i32)
    (call $log_i32 (local.get $n))
    (call $log_i32 (i32.add (local.get $i) (i32.const 1)))
  )
)

const {instance} = await WebAssembly.instantiate(buffer, {
  env: { log_i32: console.log }
});
const e = instance.exports;
console.log(e.foo(10)); // => undefined; 10 11
```

Память

```
(module
  (memory $memory 1) ;; 1 x 64 KiB
  (data (i32.const 0) "cppusrgrp") ;; offset 0
  (export "memory" (memory $memory))
  (export "capitalize" (func $capitalize))
  (func $capitalize (param $offset i32) (param $len i32)
    (local $char i32)  (local $addr i32)
    (block (loop
      (br_if 1 (i32.eqz (local.get $len)))
      (local.set $len (i32.sub (local.get $len) (i32.const 1)))
      (local.set $addr (i32.add (local.get $offset) (local.get $len)))
      (local.set $char (i32.load8_u (local.get $addr)))
      (i32.store8 (local.get $addr)
        (i32.and (local.get $char) (i32.const 0xdf)))
      (br 0)
    ))))
e.capitalize();
console.log(new TextDecoder().decode(new Uint8Array(e.memory.buffer, 0, 6))) // CPUSRGRP
```

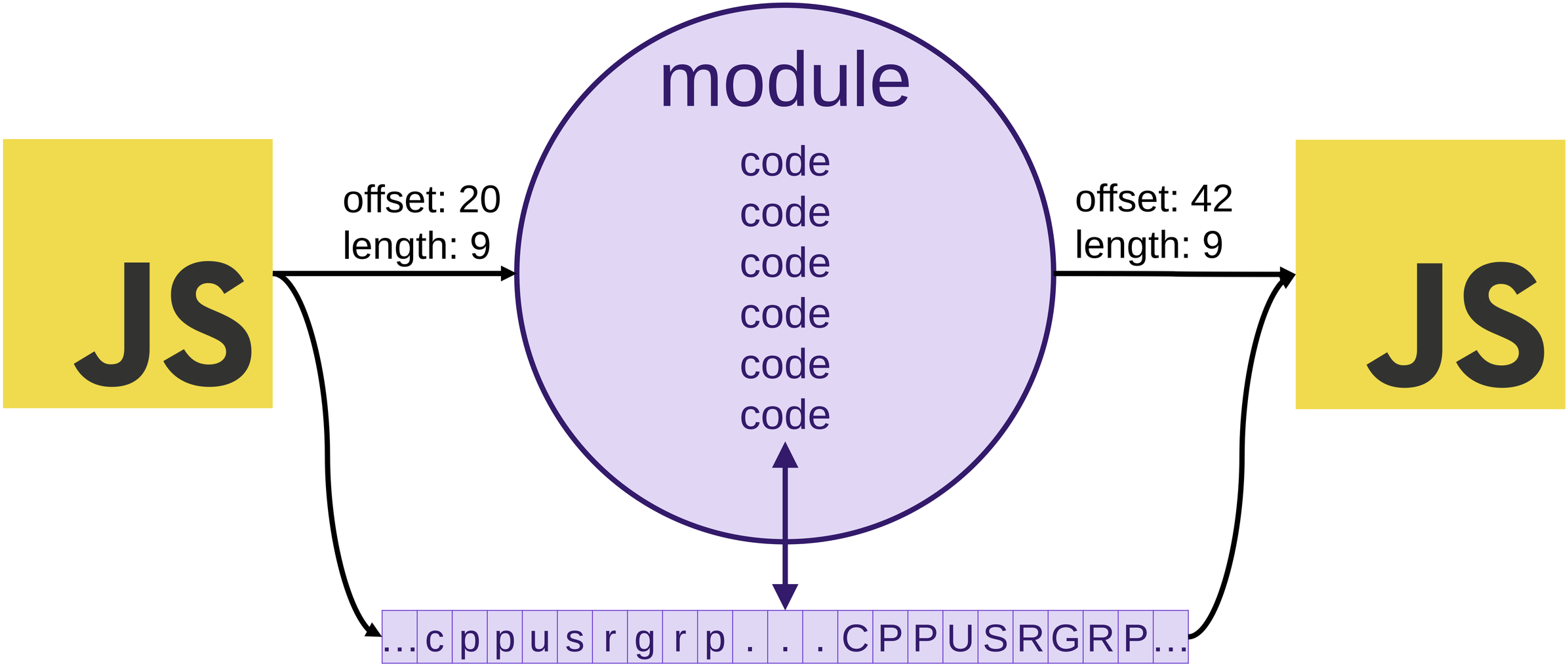
Память

```
(module
  (memory $memory 1) ;; 1 x 64 KiB
  (data (i32.const 0) "cppusrgrp") ;; offset 0
  (export "memory" (memory $memory))
  (export "capitalize" (func $capitalize))
  (func $capitalize (param $offset i32) (param $len i32)
    (local $char i32)  (local $addr i32)
    (block (loop
      (br_if 1 (i32.eqz (local.get $len)))
      (local.set $len (i32.sub (local.get $len) (i32.const 1)))
      (local.set $addr (i32.add (local.get $offset) (local.get $len)))
      (local.set $char (i32.load8_u (local.get $addr)))
      (i32.store8 (local.get $addr)
        (i32.and (local.get $char) (i32.const 0xdf)))
      (br 0)
    ))))
e.capitalize();
console.log(new TextDecoder().decode(new Uint8Array(e.memory.buffer, 0, 6))) // CPUSRGRP
```

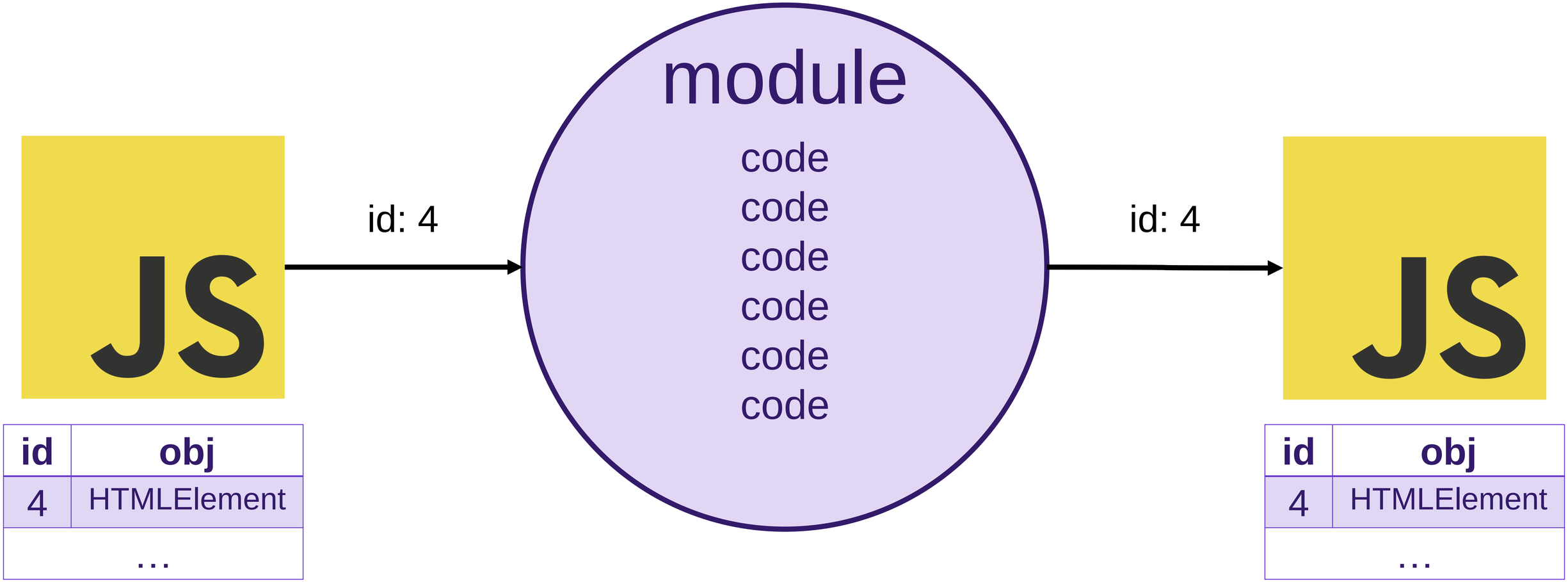
Память

```
(module
  (memory $memory 1) ;; 1 x 64 KiB
  (data (i32.const 0) "cppusrgrp") ;; offset 0
  (export "memory" (memory $memory))
  (export "capitalize" (func $capitalize))
  (func $capitalize (param $offset i32) (param $len i32)
    (local $char i32)  (local $addr i32)
    (block (loop
      (br_if 1 (i32.eqz (local.get $len)))
      (local.set $len (i32.sub (local.get $len) (i32.const 1)))
      (local.set $addr (i32.add (local.get $offset) (local.get $len)))
      (local.set $char (i32.load8_u (local.get $addr)))
      (i32.store8 (local.get $addr)
        (i32.and (local.get $char) (i32.const 0xdf)))
      (br 0)
    ))))
e.capitalize();
console.log(new TextDecoder().decode(new Uint8Array(e.memory.buffer, 0, 6))) // CPUSRGRP
```

Не примитивные типы



Не примитивные типы



id	obj
4	HTMLElement
…	

id	obj
4	HTMLElement
…	

Косвенные вызовы

```
(module
  (table $table 128 funcref)
  (type $signature (func (param i32)))
  (export "table" (table $table))
  (export "foo" (func $foo))
  (func $foo
    (call_indirect (type $signature) ;; table[index](42)
      (i32.const 42)
      (local.get $offset) ;; index
    )))

e.table.set(0, someModule.qux);
e.foo(); // someModule.qux(42)
```

Производительность

`local.set, local.get`	`mov, push, pop`
`i32.load, i32.store, …`	`mov, push, pop`
`i32.add, i32.sub, …`	`add, sub, …`
`i32.eq, i32.lt, i32.gt, …`	`test, cmp, …`
`if, else, end`	`jz, jnz, jl, jle, …`
`block, loop, br, br_if`	`jmp, jz, jnz, jl, jle, …`
`call`	`call`

Бинарный формат

```
if   (result i32)
0x04 0x7F
i32.const 42
0x41       0x2A
i32.const 255
0x41       0x01 0xff
i32.add
0x6A
end
0x0B
```

https://webassembly.github.io/spec/core/binary/instructions.html

Бинарный формат

```
(module ;; "\0asm" version
0x00 0x61 0x73 0x6D 0x01 0x00 0x00 0x00
  ;; type section length = 5 bytes
  0x01 0x05
  (type $type0 (func  (param i32)    )
  0x02          0x60  0x01  0x7F  0x00
  ;; import section length = 13 bytes
  0x02 0x0D
  ...
)
```

https://webassembly.github.io/spec/core/binary/modules.html

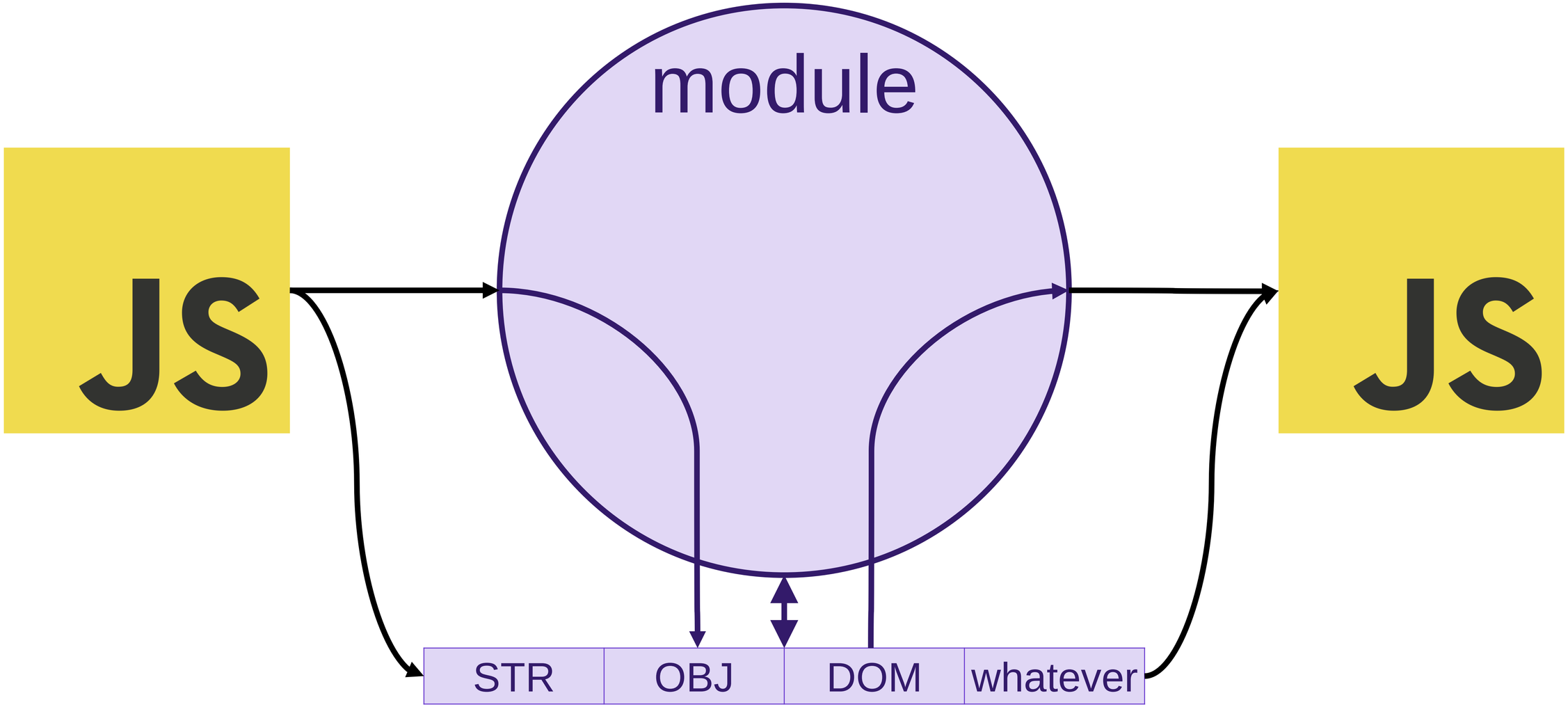
Будущее

Версионирование

- WebAssembly 1.0

- Расширения и feature-detection
  - `try { compile(); supports = run() === expected; } catch { supports = false; }`
  - Возможно будет в самом wasm файле WebAssembly/design#1173

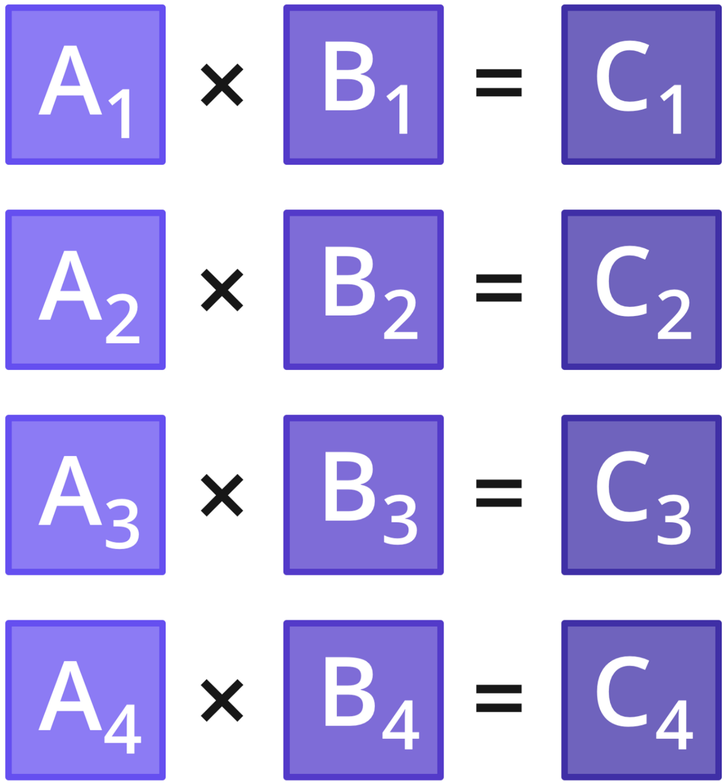
Interface types (anyref)



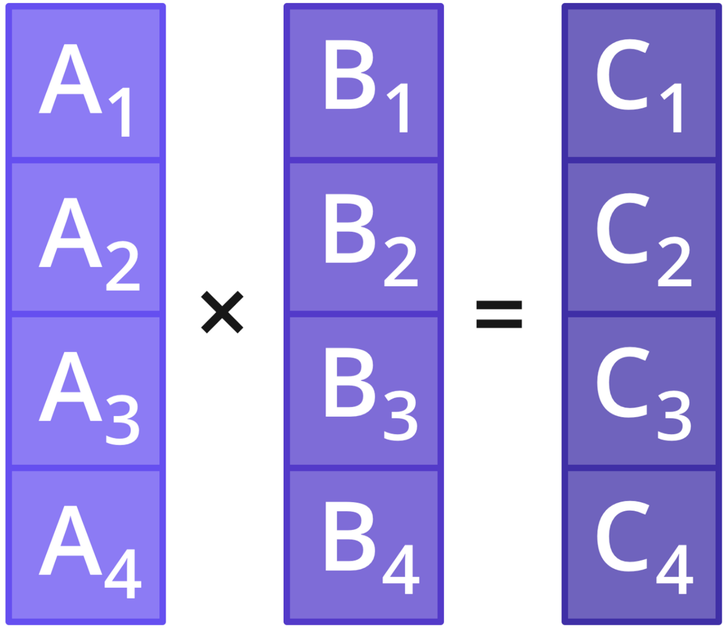
https://hacks.mozilla.org/2019/08/webassembly-interface-types/
https://github.com/WebAssembly/reference-types

SIMD

Scalar Operation

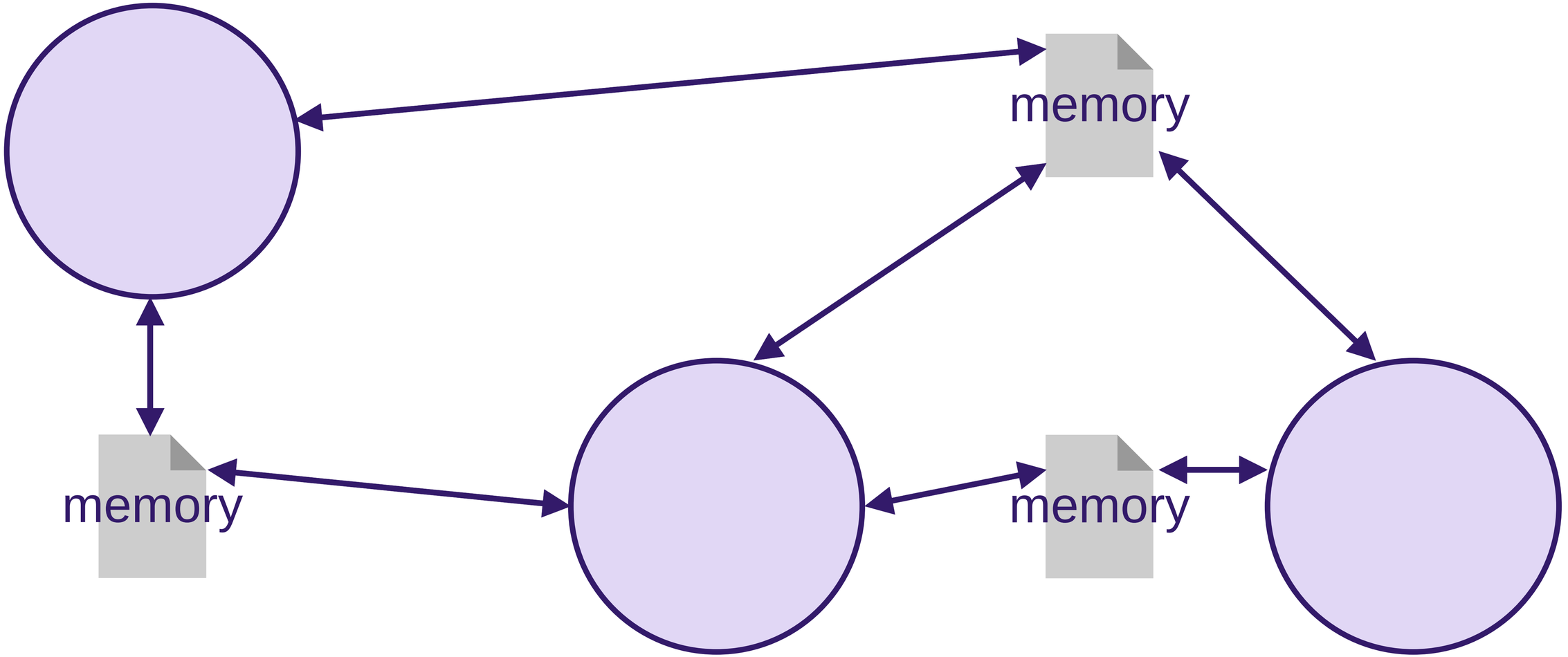


SIMD Operation



https://medium.com/wasmer/webassembly-and-simd-13badb9bf1a8
https://github.com/WebAssembly/simd/blob/master/proposals/simd/SIMD.md

Multiple memories



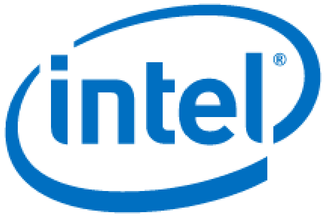
https://github.com/WebAssembly/multi-memory

И это еще не всё

- Exceptions WebAssembly/exception-handling
- Bulk operations (`memset`, `memcpy`) WebAssembly/bulk-memory-operations
- Threads (shared memory, atomics) WebAssembly/threads
- Garbage Collection WebAssembly/gc
- …
- И еще куча всего WebAssembly/proposals

Bytecode Alliance

The Bytecode Alliance is an open source community dedicated to creating secure new software foundations, building on standards such as WebAssembly and WebAssembly System Interface (WASI).



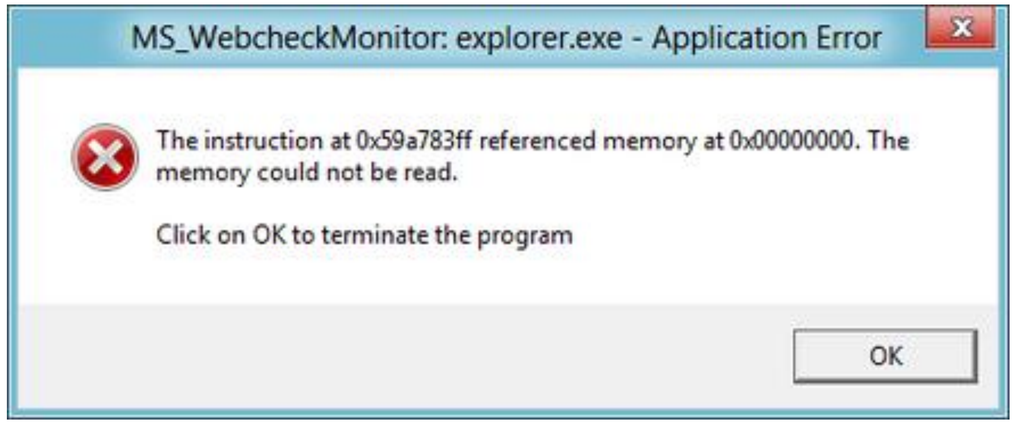
https://hacks.mozilla.org/2019/11/announcing-the-bytecode-alliance/
https://bytecodealliance.org/

FAQ

Про всякое

- У меня IE11, есть полифил?
- Можно скомпилировать JavaScript в wasm и сделать сайт быстрее?
- Можно внутри wasm скомпилировать wasm и сделать JIT?
- Есть ли string pool, кеш модулей и еще что-нибудь из jvm/.net?
- Что с отладкой?

Про безопасность



- Можно ли сделать SEGFAULT?
- Можно ли перезаписать адрес возврата?
- Можно ли перезаписать код?
- Можно ли получить доступ к файловой системе или сети?
- Aaaaa, blazor грузит dll – это новый activex!

Про безопасность

> Худшее зло, которое вы можете причинить, – нагреть комнату своим неэффективным кодом.

Про языки

- C/C++: emscripten, LLVM
- Rust: rustc + bindgen, LLVM
- C#: Mono, Blazor
- Go
- AssemblyScript

Начиная с LLVM 8.0 wasm перестал быть experimental target.

https://github.com/appcypher/awesome-wasm-langs

Не только web

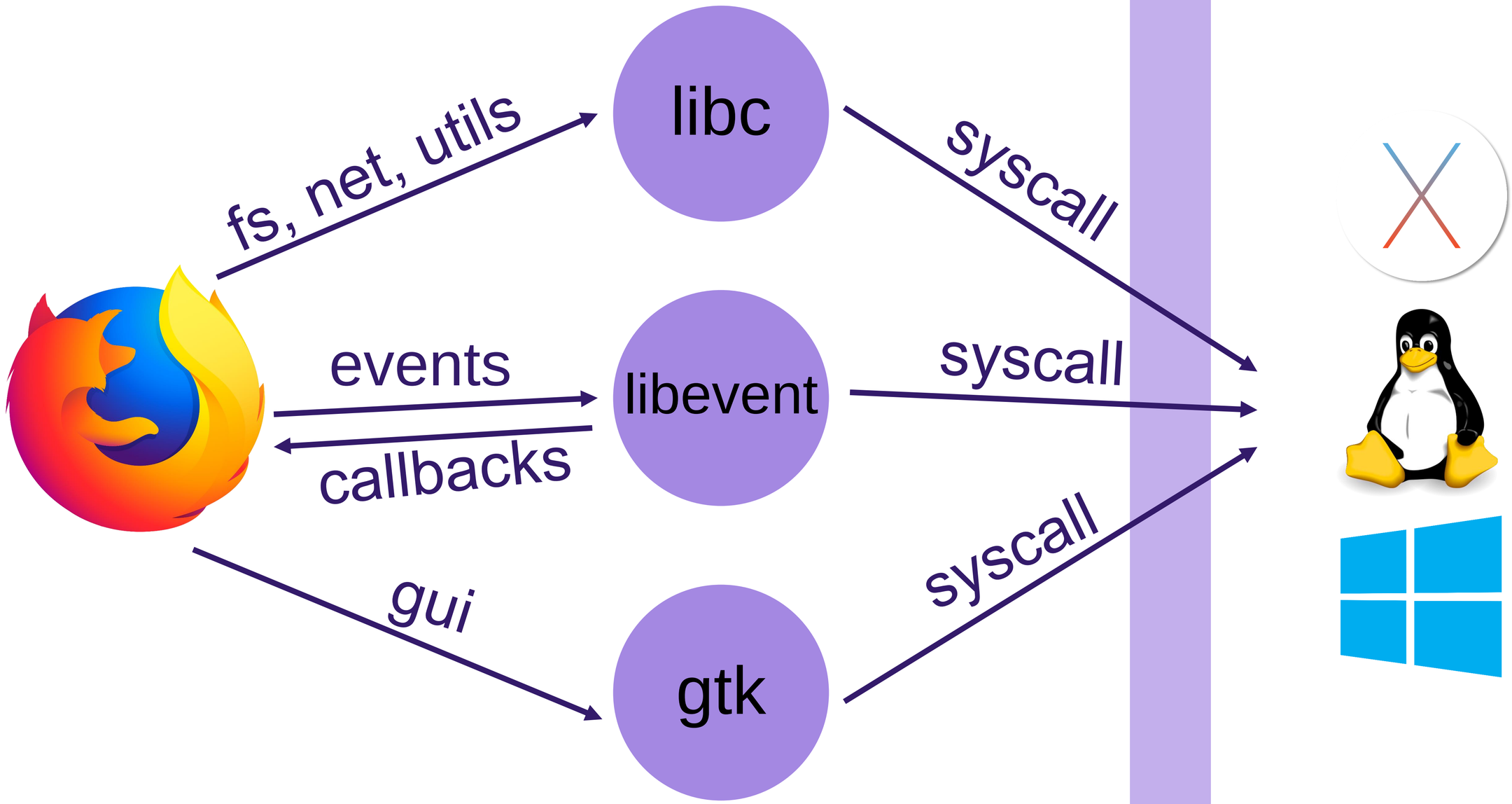
Текущие реализации

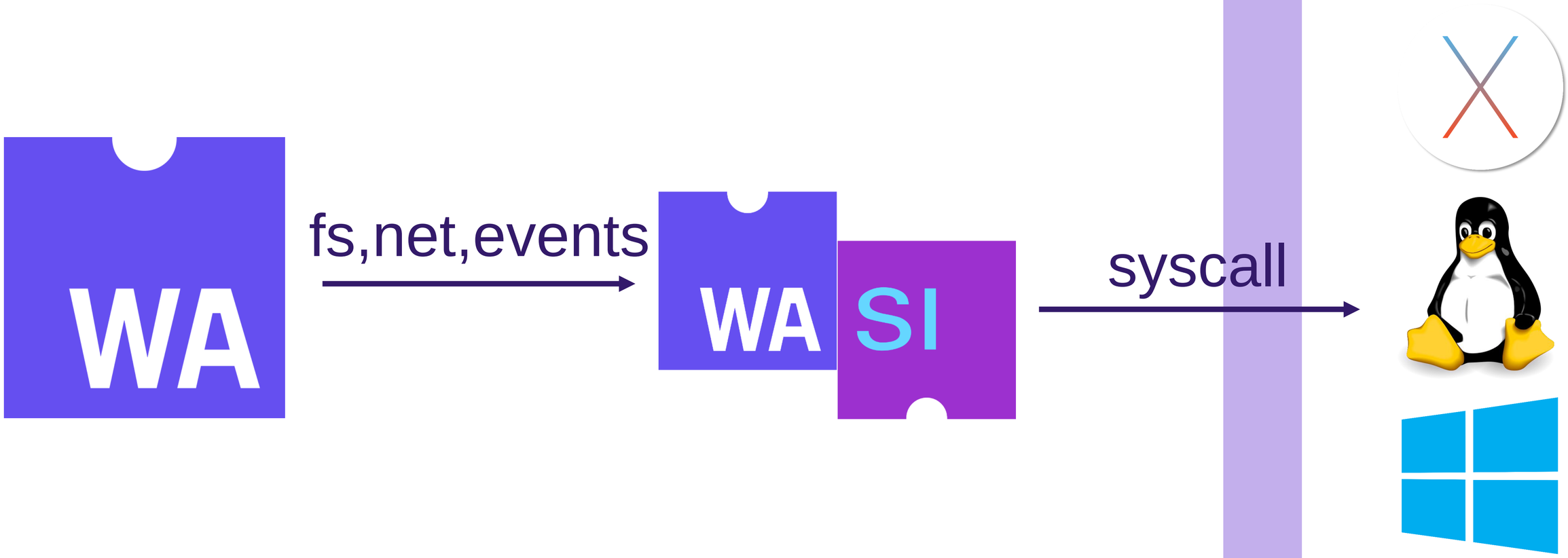
Браузерные движки:

- v8
- SpiderMonkey
- ChakraCore
- JavaScriptCore

Самостоятельные движки:

- binaryen (interpreter + js polyfill + tools)
- wasmer (+wasi)
- wasmtime (wasm meetup #2)
- wasm3


libc
fs, net, utils
syscall
events
libevent
callbacks
syscall
gui
gtk
syscall


WA
fs,net,events
WA
SI
syscall

Заключение

Где вы можете применить это у себя

- Портировать библиотеку и софт
- Числодробилки: фото/видео редакторы, игры
- Использовать один код на бэке и фронте
- Перенос экспертизы ваших коллег из соседней области (blazor)

- Легкая виртуальная машина для различных целей, например, для блокчейнов (wasm meetup #1)
- Figma делала js плагины собирая Duktape и QuickJS в wasm https://www.figma.com/blog/how-we-built-the-figma-plugin-system/

- https://webassembly.org/docs/use-cases/

Полезные инструменты

- WasmExplorer
- WasmFiddle
- WasmCodeExplorer
- WebAssembly Studio

Что почитать

- https://webassembly.github.io/spec/core/index.html
- https://hacks.mozilla.org/category/webassembly/

- WebAssembly — русскоговорящее сообщество https://t.me/WebAssembly_ru

Яндекс Карты

flapenguin.me/talks/wasm-uncut-cxxusrgrp

Яндекс Карты

Спасибо

Роенко Андрей

Руководитель группы разработки API Яндекс.Карт

flapenguin@yandex.ru

@flapenguin

flapenguin

flapenguin.me

flapenguin.me/talks/wasm-uncut-cxxusrgrp