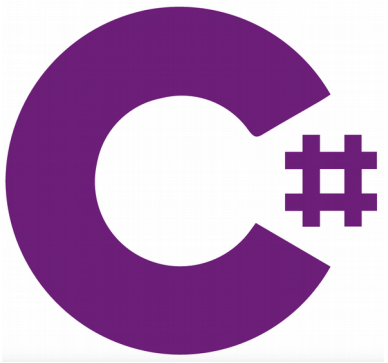
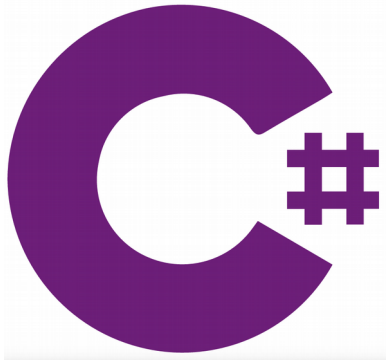
C# в браузере



+

C# в браузере



+

C# in WebAssembly:

```
public static HtmlElement TextBox { get; set; }
public static void OnClick() =>
   HtmlPage.Window.Alert("Clicked, text: " + TextBox.Invoke("value"));

public static void Main()
{
  Console.WriteLine("Hello world!");
  Console.WriteLine("Running from " + HtmlPage.BrowserInformation.UserAgent);
  Console.Error.WriteLine("Some error");

  TextBox = HtmlPage.Document.CreateElement("input");
  TextBox.SetAttribute("value", "Some text");
  HtmlPage.Document.GetElementById("app").AppendChild(TextBox);

  var btn = HtmlPage.Document.CreateElement("button");
  btn.InnerText = "Click me!";
  HtmlPage.Document.GetElementById("app").AppendChild(btn);

  JsObject.JsEval($@"{btn.JsExpr()}.onclick = function(){{
  MonoRuntime.call_by_name('SimpleWasm', 'Program', 'OnClick');
}}; ");
}
```


C# in WebAssembly:

```
public static HtmlElement TextBox { get; set; }
public static void OnClick() =>
   HtmlPage.Window.Alert("Clicked, text: " + TextBox.Invoke("value"));

public static void Main()
{
  Console.WriteLine("Hello world!");
  Console.WriteLine("Running from " + HtmlPage.BrowserInformation.UserAgent);
  Console.Error.WriteLine("Some error");

  TextBox = HtmlPage.Document.CreateElement("input");
  TextBox.SetAttribute("value", "Some text");
  HtmlPage.Document.GetElementById("app").AppendChild(TextBox);

  var btn = HtmlPage.Document.CreateElement("button");
  btn.InnerText = "Click me!";
  HtmlPage.Document.GetElementById("app").AppendChild(btn);

  JsObject.JsEval($@"{btn.JsExpr()}.onclick = function(){{
  MonoRuntime.call_by_name('SimpleWasm', 'Program', 'OnClick');
}}; ");
}
```


C# in WebAssembly:

```
public static HtmlElement TextBox { get; set; }
public static void OnClick() =>
   HtmlPage.Window.Alert("Clicked, text: " + TextBox.Invoke("value"));

public static void Main()
{
  Console.WriteLine("Hello world!");
  Console.WriteLine("Running from " + HtmlPage.BrowserInformation.UserAgent);
  Console.Error.WriteLine("Some error");

  TextBox = HtmlPage.Document.CreateElement("input");
  TextBox.SetAttribute("value", "Some text");
  HtmlPage.Document.GetElementById("app").AppendChild(TextBox);

  var btn = HtmlPage.Document.CreateElement("button");
  btn.InnerText = "Click me!";
  HtmlPage.Document.GetElementById("app").AppendChild(btn);

  JsObject.JsEval($@"{btn.JsExpr()}.onclick = function(){{
  MonoRuntime.call_by_name('SimpleWasm', 'Program', 'OnClick');
}}; ");
}
```

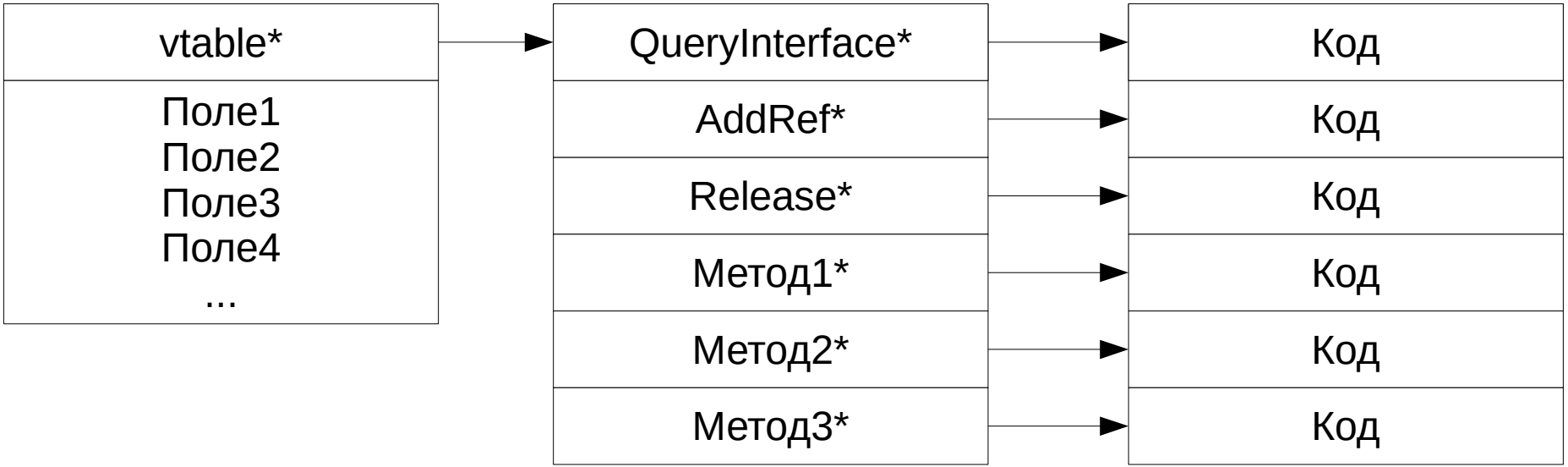
C# in WebAssembly:

```
public static HtmlElement TextBox { get; set; }
public static void OnClick() =>
   HtmlPage.Window.Alert("Clicked, text: " + TextBox.Invoke("value"));

public static void Main()
{
  Console.WriteLine("Hello world!");
  Console.WriteLine("Running from " + HtmlPage.BrowserInformation.UserAgent);
  Console.Error.WriteLine("Some error");

  TextBox = HtmlPage.Document.CreateElement("input");
  TextBox.SetAttribute("value", "Some text");
  HtmlPage.Document.GetElementById("app").AppendChild(TextBox);

  var btn = HtmlPage.Document.CreateElement("button");
  btn.InnerText = "Click me!";
  HtmlPage.Document.GetElementById("app").AppendChild(btn);

  JsObject.JsEval($@"{btn.JsExpr()}.onclick = function(){{
  MonoRuntime.call_by_name('SimpleWasm', 'Program', 'OnClick');
}}; ");
}
```

Краткая история исполнения вменяемых языков в браузере

- 1996. Microsoft анонсирует ActiveX

Component Object Model



Взаимодействие на *уровне объектов* *любых* языков с *любыми*

Краткая история исполнения вменяемых языков в браузере

Краткая история исполнения вменяемых языков в браузере

- 1996. Microsoft анонсирует ActiveX

Краткая история исполнения вменяемых языков в браузере

- 1996. Microsoft анонсирует ActiveX
- 2000. В Macromedia Flash появляется поддержка ActionScript

Краткая история исполнения вменяемых языков в браузере

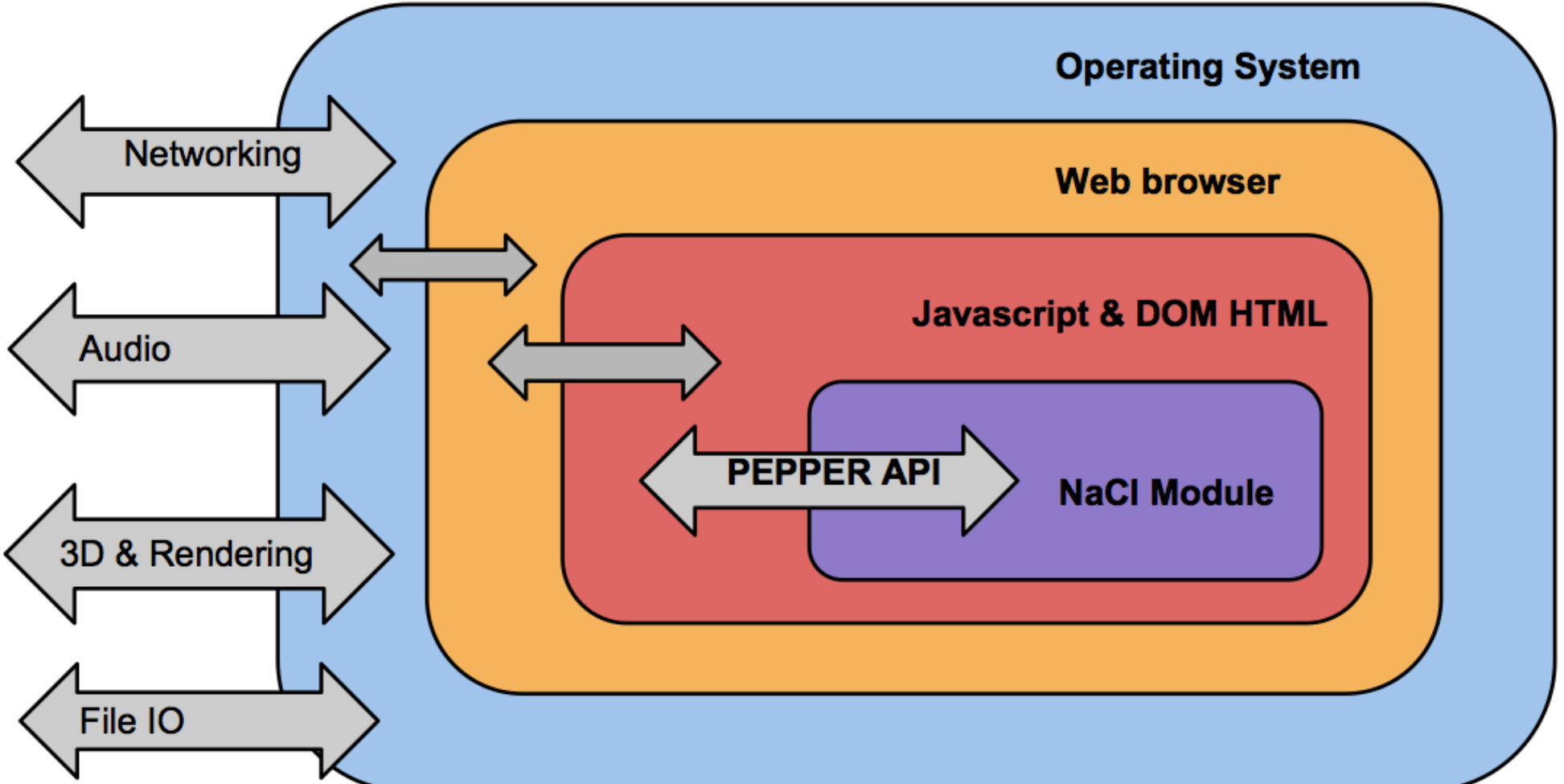
- 1996. Microsoft анонсирует ActiveX
- 2000. В Macromedia Flash появляется поддержка ActionScript
- 2007. Microsoft выпускает Silverlight под Windows и OSX специальным рантаймом *CoreCLR*

Краткая история исполнения вменяемых языков в браузере

- 1996. Microsoft анонсирует ActiveX
- 2000. В Macromedia Flash появляется поддержка ActionScript
- 2007. Microsoft выпускает Silverlight под Windows и OSX специальным рантаймом *CoreCLR*
- 2009. Силами проекта Mono создаётся Moonlight под Linux

Краткая история исполнения вменяемых языков в браузере

- 1996. Microsoft анонсирует ActiveX
- 2000. В Macromedia Flash появляется поддержка ActionScript
- 2007. Microsoft выпускает Silverlight под Windows и OSX специальным рантаймом *CoreCLR*
- 2009. Силами проекта Mono создаётся Moonlight под Linux
- 2011. Google выпускает технологию **Na**tive**Cl**ient: OpenGL ES 2.0, нативный код (или JIT из байткода), многопоточность, нормальные указатели, можно запускать Mono и C#.


Operating System
Networking
Web browser
Audio
Javascript & DOM HTML
PEPPER API
NaCl Module
3D & Rendering
File IO

Плюшки NaCl

- Сэндбоксинг и безопасность
- Нормальный доступ к инструкциям процессора и памяти, возможна реализация JIT
- Поддержка многопоточности

Краткая история исполнения вменяемых языков в браузере

- 2013. Появление Javascript Typed Arrays, asm.js, развитие тулчейна emscripten

```
int fib(int n)
{
   if(n == 0)
      return 0;
   if(n == 1)
      return 1;
   return fib(n - 1) + fib(n - 2);
}
```

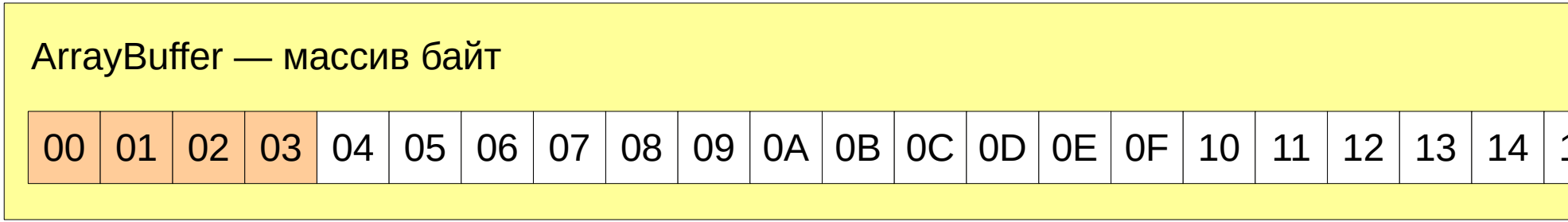
```
int fib(int n)
{
   if(n == 0)
      return 0;
   if(n == 1)
      return 1;
   return fib(n - 1) + fib(n - 2);
}
```

```
function fib(n)
{
   if(n == 0)
      return 0;
   if(n == 1)
      return 1;
   return ((fib(n-1)) + (fib(n-2)));
}
```

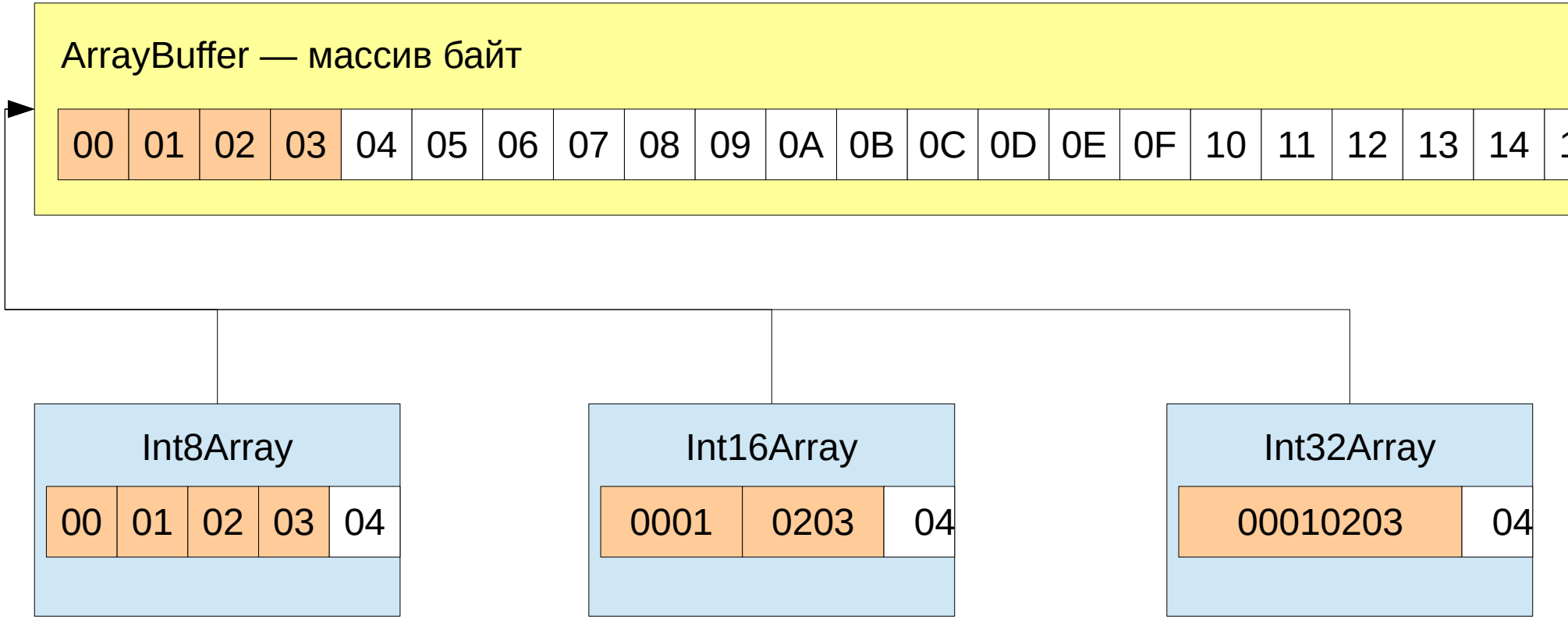

```
int fib(int n)
{
   if(n == 0)
      return 0;
   if(n == 1)
      return 1;
   return fib(n - 1) + fib(n - 2);
}
```

```
function fib(n)
{
   if(n == 0)
      return 0;
   if(n == 1)
      return 1;
   return ((fib(n-1)) + (fib(n-2)));
}

function fib2(n)
{
   n = n | 0; // это int32, правда-правда
   if(n == 0)
      return 0 | 0;
   if(n == 1)
      return 1 | 0;
   return ((fib(n-1) | 0) + (fib(n-2) | 0)) | 0;
}
```

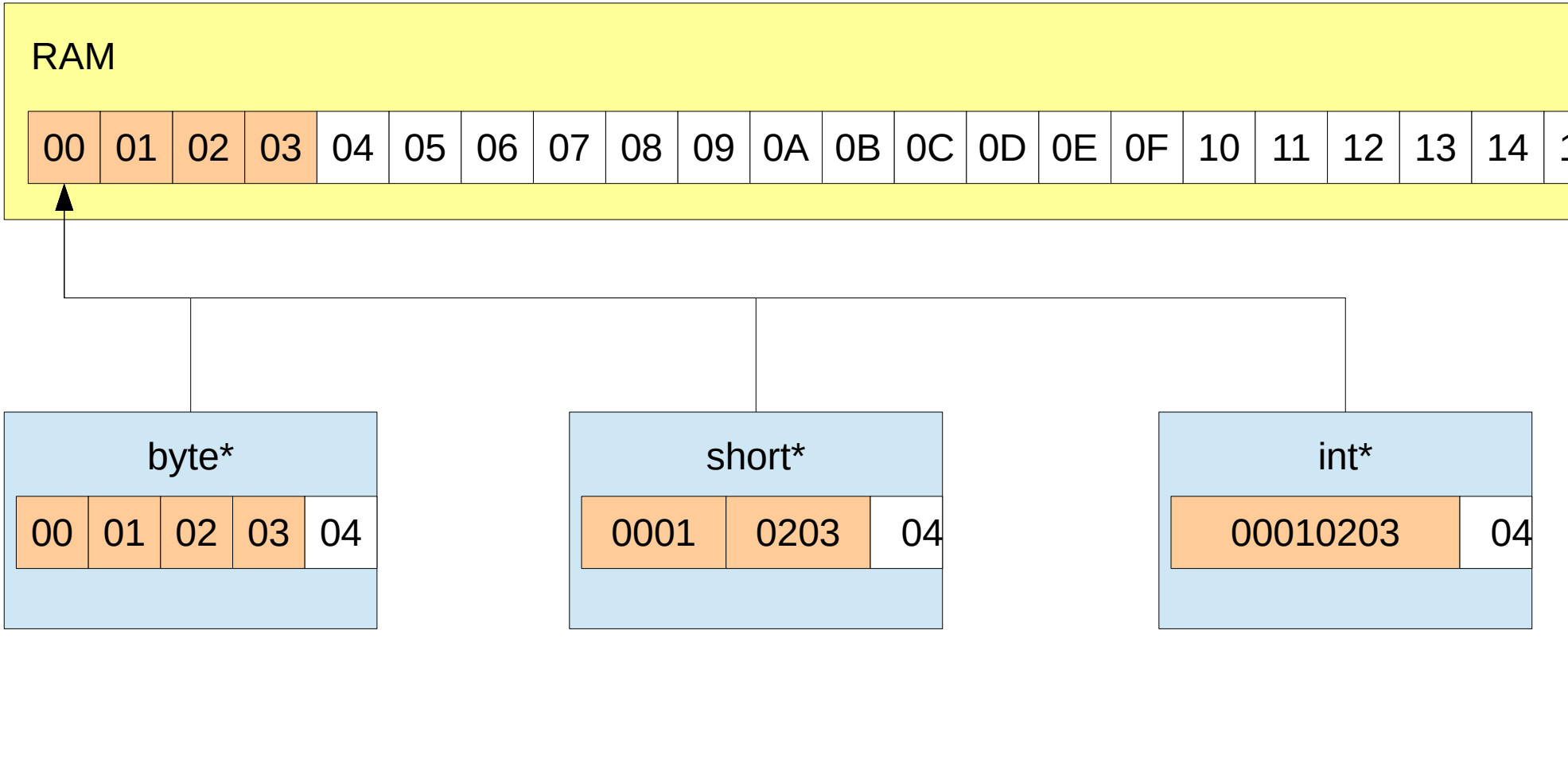

Javascript TypedArray



Javascript TypedArray



Низкоуровневая модель памяти компьютеров



```
int mystrcmp(
    char* s1, char* s2)
{
   for(size_t c = 0; ;c++)
   {
       int d = s1[c] - s2[c];
       if(d != 0)
           return d;
       if(s1[c] == 0)
           return 0;
   }
}
```


```
int mystrcmp(
    char* s1, char* s2)
{
   for(size_t c = 0; ;c++)
   {
       int d = s1[c] - s2[c];
       if(d != 0)
           return d;
       if(s1[c] == 0)
           return 0;
   }
}
```

```
function mystrcmp(s1, s2)
{
   s1 = s1 | 0;
   s2 = s2 | 0;
   for(var c = 0 | 0; ;c++)
   {
       var d = ( HEAP8[(s1 + c) | 0] -
                 HEAP8[(s2 + c) | 0] ) | 0;
       if(d != 0)
           return d | 0;
       if(HEAP8[(s1 + c) | 0] == 0)
           return 0 | 0;
   }
}
```


```
function _mystrcmp($0,$1) {
 $0 = $0|0;
 $1 = $1|0;
 var $10 = 0, $11 = 0, $12 = 0, $13 = 0, $14 = 0, $15 = 0, $16 = 0, $17 = 0, $18 = 0, $19 = 0, $2 = 0, $20 = 0, $21 = 0, $22 = 0, $23 = 0, $24 = 0, $25 = 0, $26 = 0, $27 = 0, $28 = 0;
 var $29 = 0, $3 = 0, $4 = 0, $5 = 0, $6 = 0, $7 = 0, $8 = 0, $9 = 0, label = 0, sp = 0;
 sp = STACKTOP;
 STACKTOP = STACKTOP + 32|0; if ((STACKTOP|0) >= (STACK_MAX|0)) abortStackOverflow(32|0);
 $3 = $0;
 $4 = $1;
 $5 = 0;
 while(1) {
  $7 = $3;
  $8 = $5;
  $9 = (($7) + ($8)|0);
  $10 = HEAP8[$9>>0]|0;
  $11 = $10 << 24 >> 24;
  $12 = $4;
  $13 = $5;
  $14 = (($12) + ($13)|0);
  $15 = HEAP8[$14>>0]|0;
  $16 = $15 << 24 >> 24;
  $17 = (($11) - ($16))|0;
  $6 = $17;
  $18 = $6;
  $19 = ($18|0)!=(0);
  if ($19) {
   label = 3;
   break;
  }
  $21 = $3;
  $22 = $5;
  $23 = (($21) + ($22)|0);
  $24 = HEAP8[$23>>0]|0;
  $25 = $24 << 24 >> 24;
  $26 = ($25|0)==(0);
  if ($26) {
   label = 5;
   break;
  }
  $27 = $5;
  $28 = (($27) + 1)|0;
  $5 = $28;
 }
 if ((label|0) == 3) {
  $20 = $6;
  $2 = $20;
  $29 = $2;
  STACKTOP = sp;return ($29|0);
 }
 else if ((label|0) == 5) {
  $2 = 0;
  $29 = $2;
  STACKTOP = sp;return ($29|0);
 }
 return (0)|0;
}
```

```
function allocateUTF8(str) {
 var size = lengthBytesUTF8(str) + 1;
 var ret = _malloc(size);
 if (ret) stringToUTF8Array(str, HEAP8, ret, size);
 return ret;
}
```


```
function stringToUTF8Array(str, outU8Array, outIdx, maxBytesToWrite) {
 if (!(maxBytesToWrite > 0)) // Parameter maxBytesToWrite is not optional. Negative values, 0, null, undefined and false each don't write
any bytes.
  return 0;
 var startIdx = outIdx;
 var endIdx = outIdx + maxBytesToWrite - 1; // -1 for string null terminator.
 for (var i = 0; i < str.length; ++i) {
  // Gotcha: charCodeAt returns a 16-bit word that is a UTF-16 encoded code unit, not a Unicode code point of the character! So decode UT
>UTF32->UTF8.
  // See http://unicode.org/faq/utf_bom.html#utf16-3
  // For UTF8 byte structure, see http://en.wikipedia.org/wiki/UTF-8#Description and https://www.ietf.org/rfc/rfc2279.txt and
https://tools.ietf.org/html/rfc3629
  var u = str.charCodeAt(i); // possibly a lead surrogate
  if (u >= 0xD800 && u <= 0xDFFF) u = 0x10000 + ((u & 0x3FF) << 10) | (str.charCodeAt(++i) & 0x3FF);
  if (u <= 0x7F) {
   if (outIdx >= endIdx) break;
   outU8Array[outIdx++] = u;
  } else if (u <= 0x7FF) {
   if (outIdx + 1 >= endIdx) break;
   outU8Array[outIdx++] = 0xC0 | (u >> 6);
   outU8Array[outIdx++] = 0x80 | (u & 63);
  } else if (u <= 0xFFFF) {
   if (outIdx + 2 >= endIdx) break;
   outU8Array[outIdx++] = 0xE0 | (u >> 12);
   outU8Array[outIdx++] = 0x80 | ((u >> 6) & 63);
   outU8Array[outIdx++] = 0x80 | (u & 63);
  } else if (u <= 0x1FFFFF) {
   if (outIdx + 3 >= endIdx) break;
   outU8Array[outIdx++] = 0xF0 | (u >> 18);
   outU8Array[outIdx++] = 0x80 | ((u >> 12) & 63);
   outU8Array[outIdx++] = 0x80 | ((u >> 6) & 63);
   outU8Array[outIdx++] = 0x80 | (u & 63);
  } else if (u <= 0x3FFFFFF) {
   if (outIdx + 4 >= endIdx) break;
   outU8Array[outIdx++] = 0xF8 | (u >> 24);
   outU8Array[outIdx++] = 0x80 | ((u >> 18) & 63);
   outU8Array[outIdx++] = 0x80 | ((u >> 12) & 63);
   outU8Array[outIdx++] = 0x80 | ((u >> 6) & 63);
   outU8Array[outIdx++] = 0x80 | (u & 63);
```

```
function _mystrcmp($0,$1) {
 $0 = $0|0;
 $1 = $1|0;
 var $10 = 0, $11 = 0, $12 = 0, $13 = 0, $14 = 0, $15 = 0, $16 = 0, $17 = 0, $18 = 0, $19 = 0, $2 = 0, $20 = 0,
$21 = 0, $22 = 0, $23 = 0, $24 = 0, $25 = 0, $26 = 0, $27 = 0, $28 = 0;
 var $29 = 0, $3 = 0, $4 = 0, $5 = 0, $6 = 0, $7 = 0, $8 = 0, $9 = 0, label = 0, sp = 0;
 sp = STACKTOP;
 STACKTOP = STACKTOP + 32|0; if ((STACKTOP|0) >= (STACK_MAX|0)) abortStackOverflow(32|0);
 $3 = $0;
 $4 = $1;
 $5 = 0;
 while(1) {
  $7 = $3;
  $8 = $5;
  $9 = (($7) + ($8)|0);
  $10 = HEAP8[$9>>0]|0;
  $11 = $10 << 24 >> 24;
  $12 = $4;
  $13 = $5;
  $14 = (($12) + ($13)|0);
  $15 = HEAP8[$14>>0]|0;
  $16 = $15 << 24 >> 24;
  $17 = (($11) - ($16))|0;
  $6 = $17;
  $18 = $6;
  $19 = ($18|0)!=(0);
```

Краткая история исполнения вменяемых языков в браузере

Краткая история исполнения вменяемых языков в браузере

- 2013. Появление Javascript Typed Arrays, asm.js, развитие тулчейна emscripten
- 2015. Разработчики браузеров наконец-то договорились, что надо с этим что-то делать.

Краткая история исполнения вменяемых языков в браузере

- 2013. Появление Javascript Typed Arrays, asm.js, развитие тулчейна emscripten
- 2015. Разработчики браузеров наконец-то договорились, что надо с этим что-то делать.
- 2017. Релиз первой версии спецификации WebAssembly и поддержка оной в распространённых браузерах

Наш int fib(int n) в байткоде WebAssembly (aka много непонятной фигни)

```
(func $_fib (; 19 ;) (param $0 i32) (result i32)
 (local $1 i32)
 (block $switch (result i32)
  (block $switch-default
   (block $switch-case0
    (block $switch-case
     (br_table $switch-case0 $switch-case $switch-default
      (get_local $0)
     )
    )
    (set_local $0
     (i32.const 1)
    )
    (return
     (i32.const 1)
    )
   )
   (set_local $0
    (i32.const 0)
   )
   (return
    (i32.const 0)
   )
  )
  (set_local $1
   (i32.add
    (get_local $0)
    (i32.const -1)
   )
  )
  (set_local $1
   (call $_fib
    (get_local $1)
   )
  )
  (set_local $0
   (i32.add
    (get_local $0)
    (i32.const -2)
   )
  )
  (set_local $0
   (call $_fib
    (get_local $0)
   )
  )
  (set_local $0
   (i32.add
    (get_local $0)
    (get_local $1)
   )
  )
  ;;@ test.c:24:0
  (get_local $0)
 )
)
```

Преимущества WebAssembly перед asm.js

Преимущества WebAssembly перед asm.js

- Бинарный формат байткода

Преимущества WebAssembly перед asm.js

- Бинарный формат байткода
- Заточен специально под «низкоуровневую» работу

Преимущества WebAssembly перед asm.js

- Бинарный формат байткода
- Заточен специально под «низкоуровневую» работу
- Поддержка невыровнянного доступа к памяти, 64-битных операций, ряд специализированных опкодов байткода, вызовы «по указателю» (call_indirect)

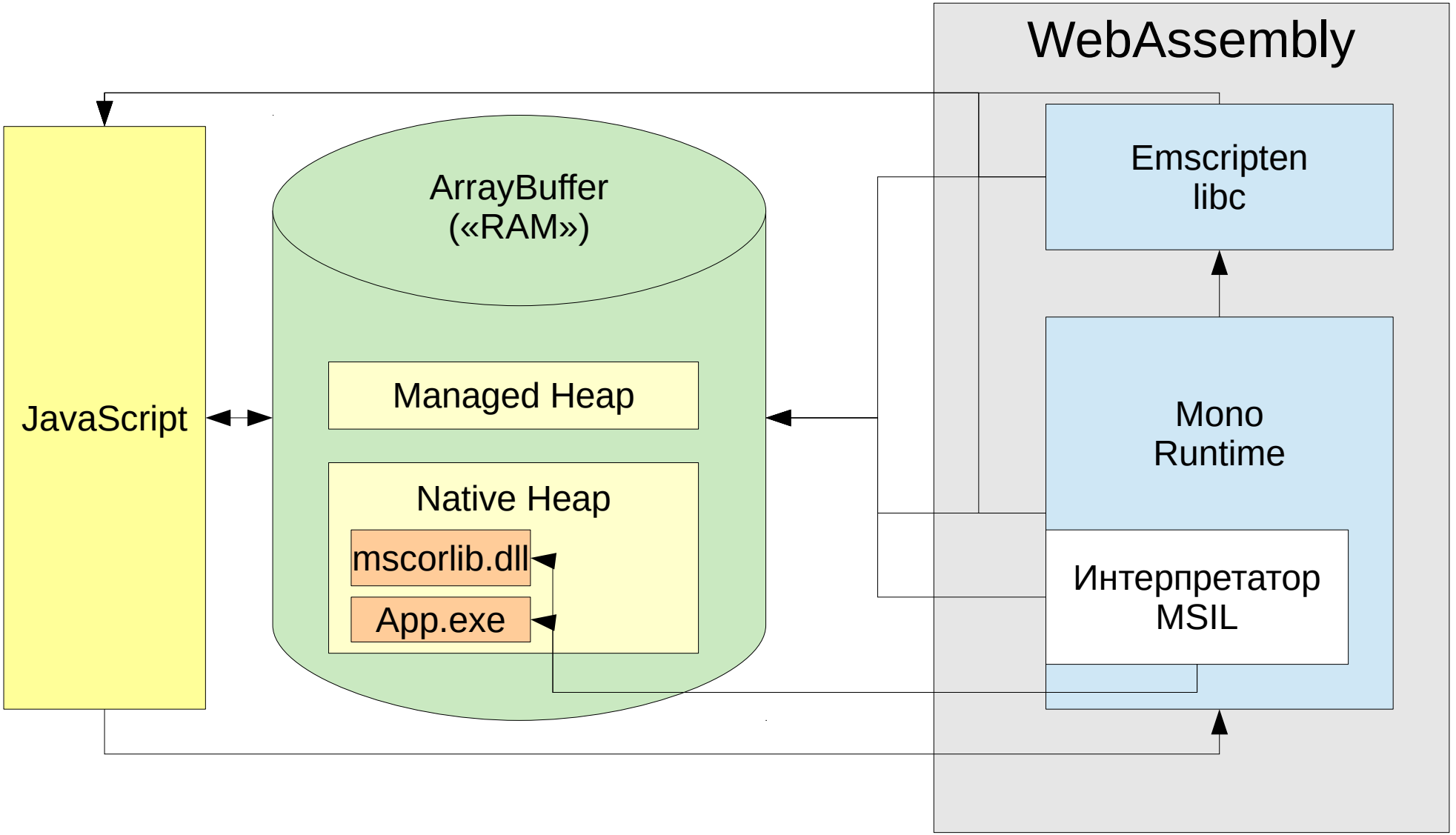
CoreCLR ...

CoreRT ...

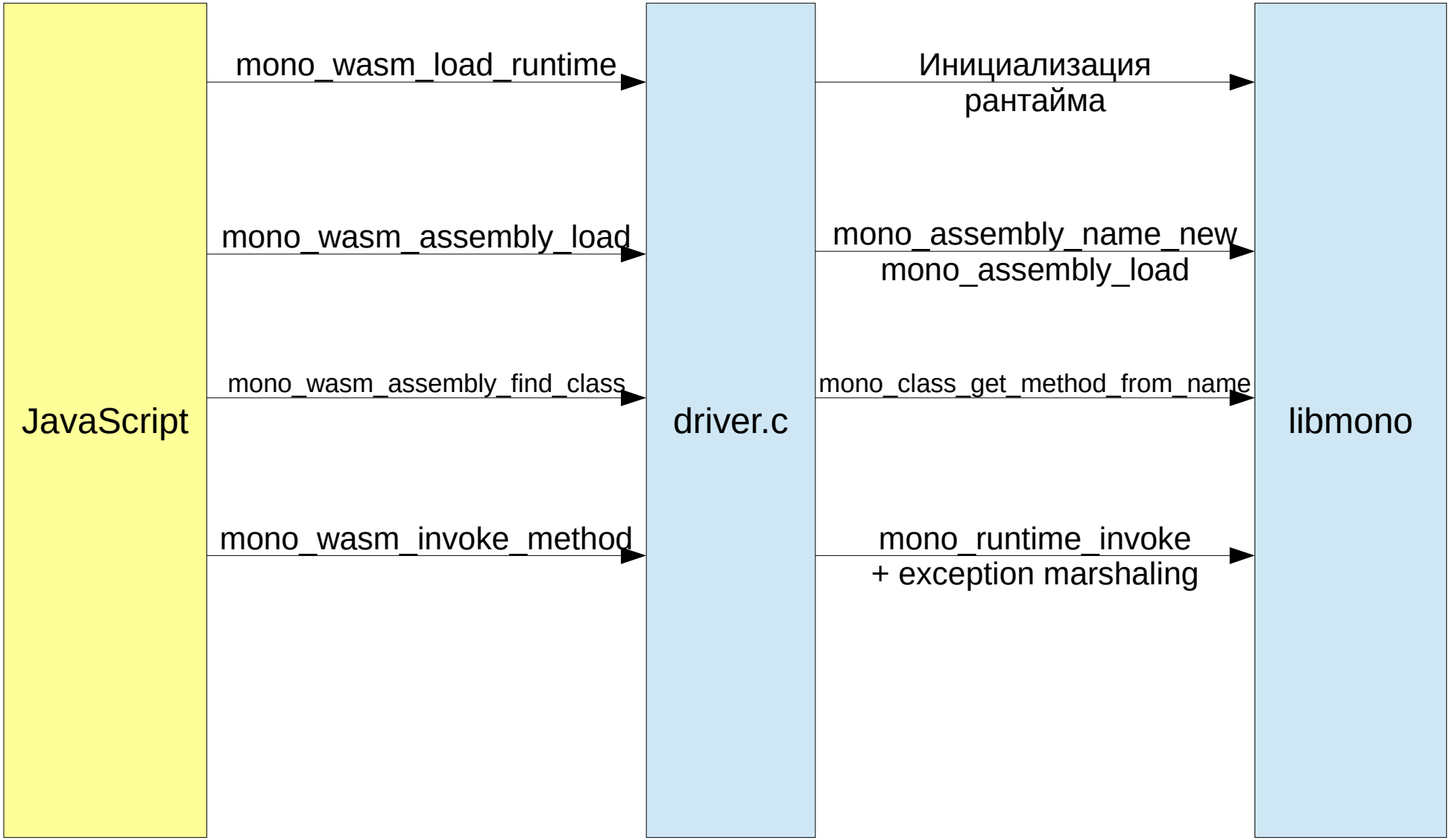
Mono ...

libmono ...

... написано на C/C++


WebAssembly
JavaScript
ArrayBuffer
(«RAM»)
Managed Heap
Native Heap
mscorlib.dll
App.exe
Emscripten
libc
Mono
Runtime
Интерпретатор
MSIL

ДЕМО


JavaScript
mono_wasm_load_runtime
mono_wasm_assembly_load
mono_wasm_assembly_find_class
mono_wasm_invoke_method
driver.c
Инициализация рантайма
mono_assembly_name_new
mono_assembly_load
mono_class_get_method_from_name
mono_runtime_invoke
+ exception marshaling
libmono

```
mergeInto(LibraryManager.library, {
  draw_wasm_image: function(ptr, width, height) {
     Module['draw_wasm_image'](ptr, width, height);
  },
});
```

```
mergeInto(LibraryManager.library, {
  draw_wasm_image: function(ptr, width, height) {
     Module['draw_wasm_image'](ptr, width, height);
  },
});

extern void draw_wasm_image(void*,int,int);
```

```
mergeInto(LibraryManager.library, {
  draw_wasm_image: function(ptr, width, height) {
     Module['draw_wasm_image'](ptr, width, height);
  },
});

extern void draw_wasm_image(void*,int,int);


mono_add_internal_call(
   "SimpleWasm.ImageRender::DrawBitmap",
   draw_wasm_image);
```

Готовность связки C#+Mono+WebAssembly к использованию

Готовность связки C#+Mono+WebAssembly к использованию

- Можно взять любую совместимую с NETStandard/Mono не порождающую потоков и не стучащуюся к сети managed-библиотеку, и она будет работать.

Готовность связки C#+Mono+WebAssembly к использованию

- Можно взять любую совместимую с NETStandard/Mono не порождающую потоков и не стучащуюся к сети managed-библиотеку, и она будет работать.
- Пока нет вменяемого SDK и тулчейна

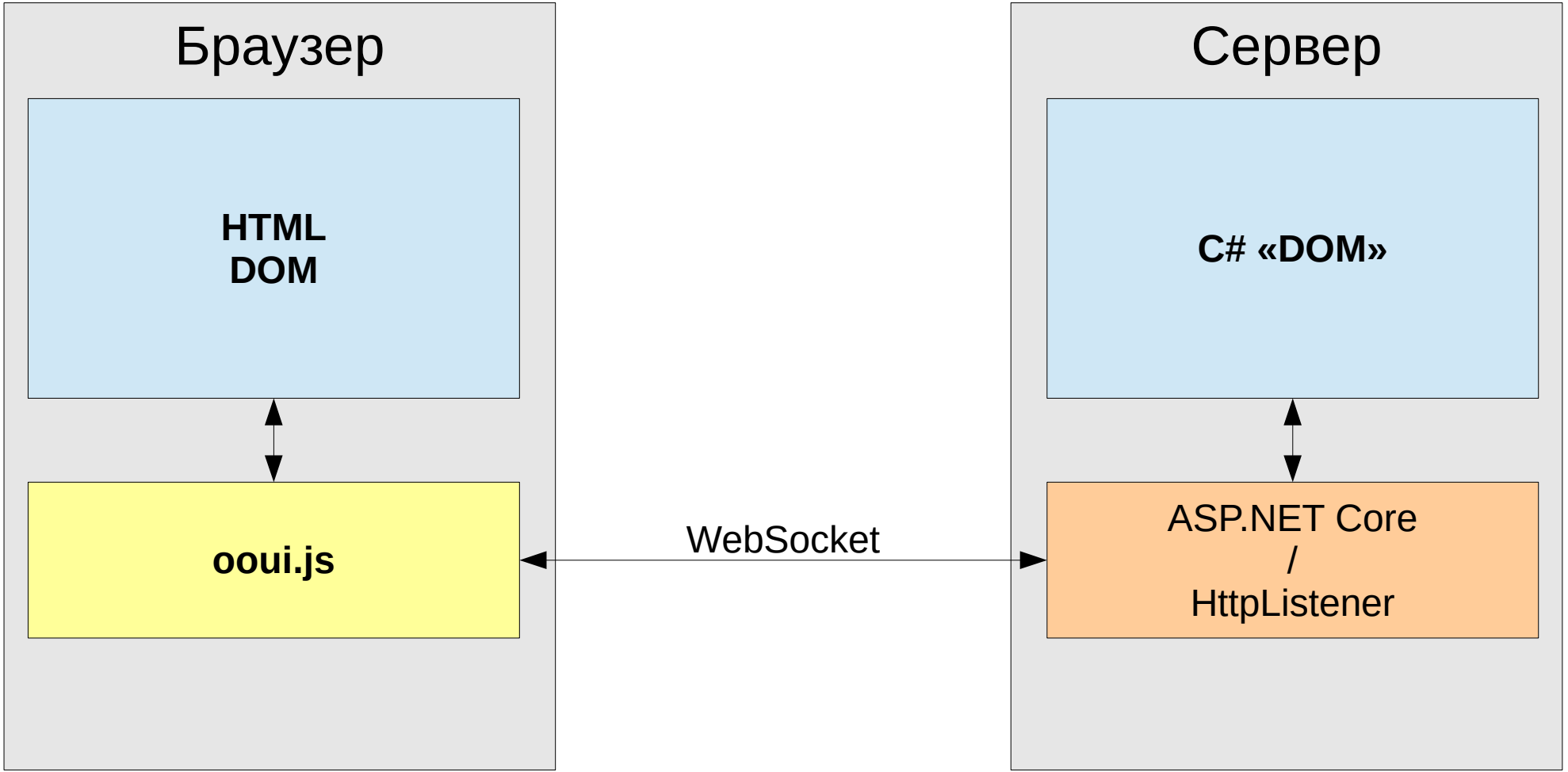
Готовность связки C#+Mono+WebAssembly к использованию

- Можно взять любую совместимую с NETStandard/Mono не порождающую потоков и не стучащуюся к сети managed-библиотеку, и она будет работать.
- Пока нет вменяемого SDK и тулчейна
- Пока нет нормального взаимодействия с JS-миром

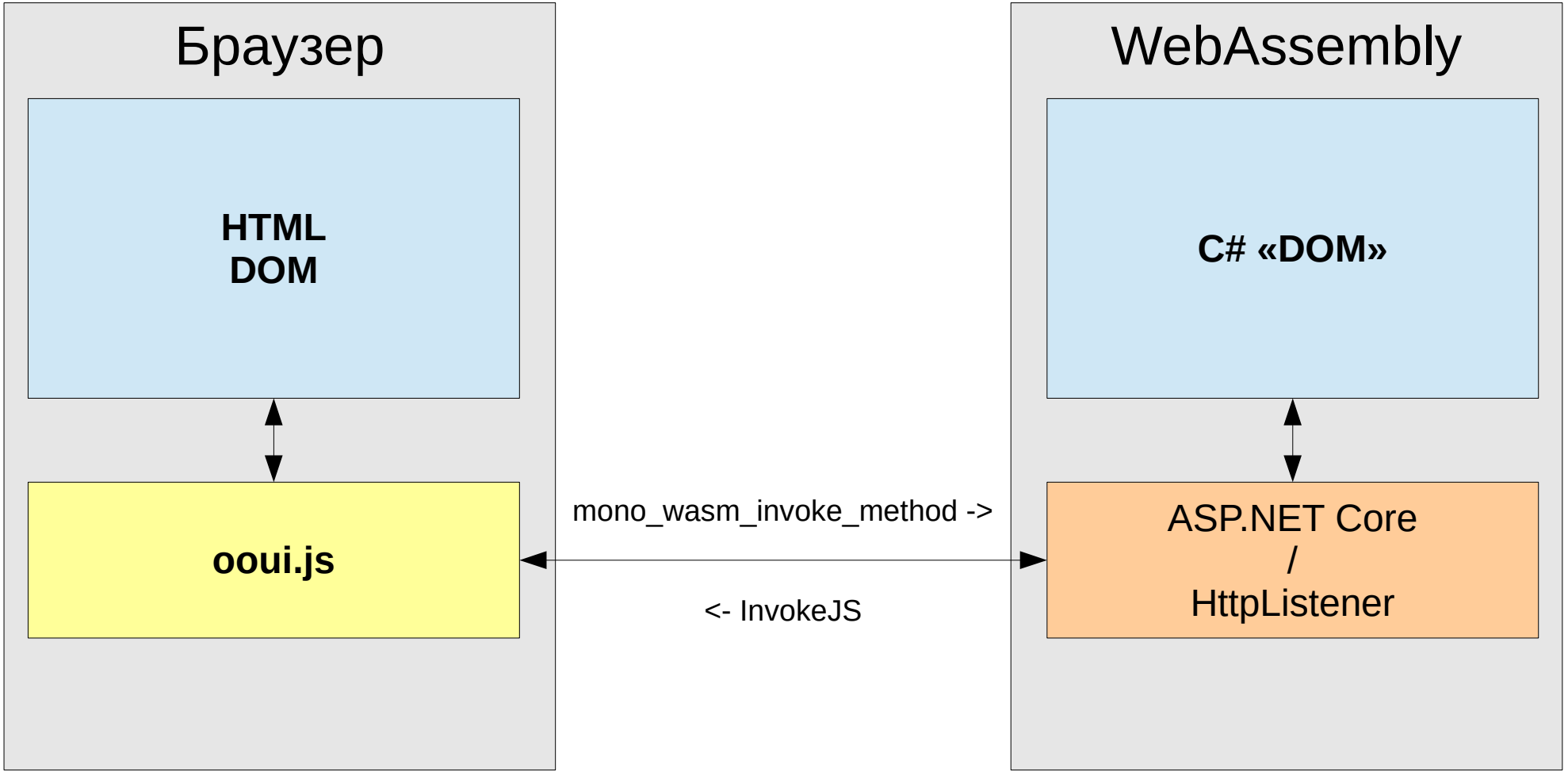
Готовность связки C#+Mono+WebAssembly к использованию

- Можно взять любую совместимую с NETStandard/Mono не порождающую потоков и не стучащуюся к сети managed-библиотеку, и она будет работать.
- Пока нет вменяемого SDK и тулчейна
- Пока нет нормального взаимодействия с JS-миром
- Пока нет компиляции C# напрямую в WebAssembly, но процесс идёт, есть работающие демки

Ooui


Браузер
HTML
DOM
ooui.js
WebSocket
Сервер
C# «DOM»
ASP.NET Core
/
HttpListener

Ooui


Браузер
HTML
DOM
ooui.js
mono_wasm_invoke_method ->
<- InvokeJS
WebAssembly
C# «DOM»
ASP.NET Core
/
HttpListener

ДЕМО

Razor здорового человека

```
<div>
    <span class="test">@Model</span>
</div>
```

Razor здорового человека

```
<div>
    <span class="test">@Model</span>
</div>
```

```
WriteLiteral("<div>\r\n    <span");
WriteLiteral(" class=\"test\"");
WriteLiteral(">");
Write(Model);
WriteLiteral("</span>\r\n</div>\r\n");
```

```
builder.AddContent(6, "    ");
builder.OpenElement(7, "center");
builder.AddContent(8, "\n        ");
builder.OpenElement(9, "div");
builder.AddAttribute(10, "style", "max-width: 600px; marg
builder.AddContent(11, "\n            ");
builder.OpenElement(12, "div");
builder.AddContent(13, Message);
builder.CloseElement();
builder.AddContent(14, "\n            ");
builder.OpenElement(15, "label");
builder.AddContent(16, "Login");
builder.CloseElement();
builder.AddContent(17, "\n            ");
builder.OpenElement(18, "input");
builder.AddAttribute(19, "class", "form-control");
builder.AddAttribute(20, "value",
Microsoft.AspNetCore.Blazor.Components.BindMethods.GetVal
builder.AddAttribute(21, "onchange",
```

Virtual DOM

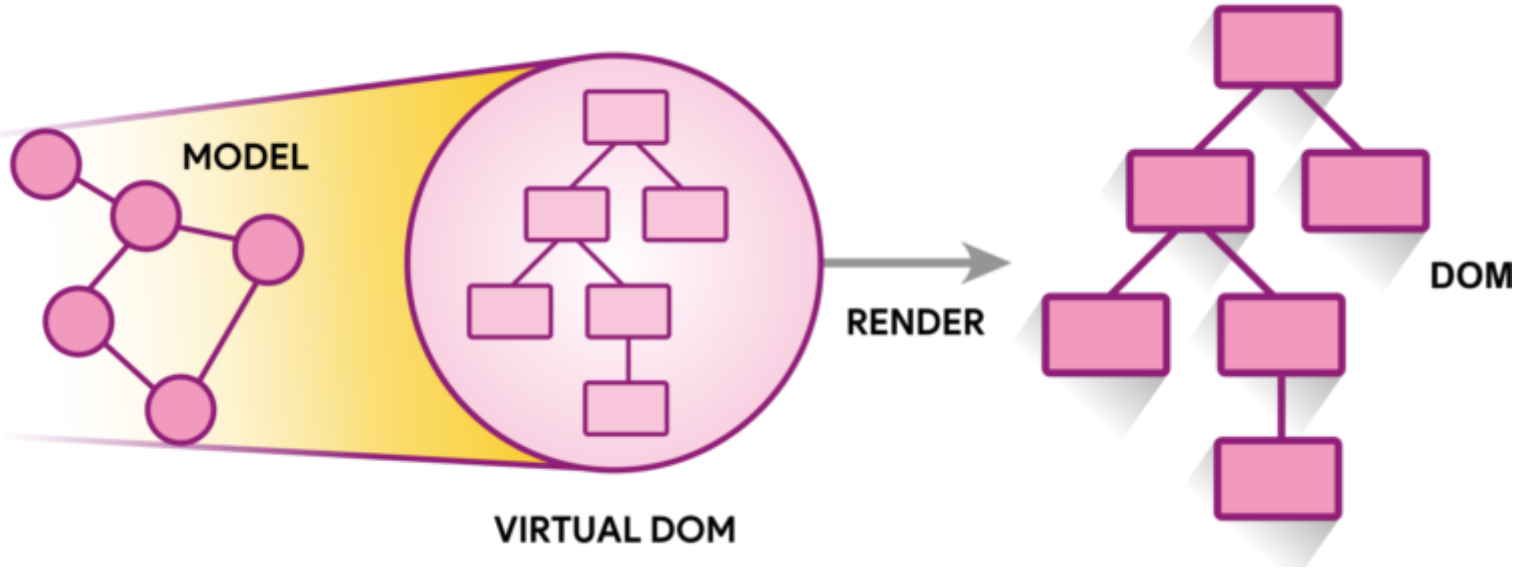


Схема из http://www.systango.com/blog/react-js-infinity/

Virtual DOM

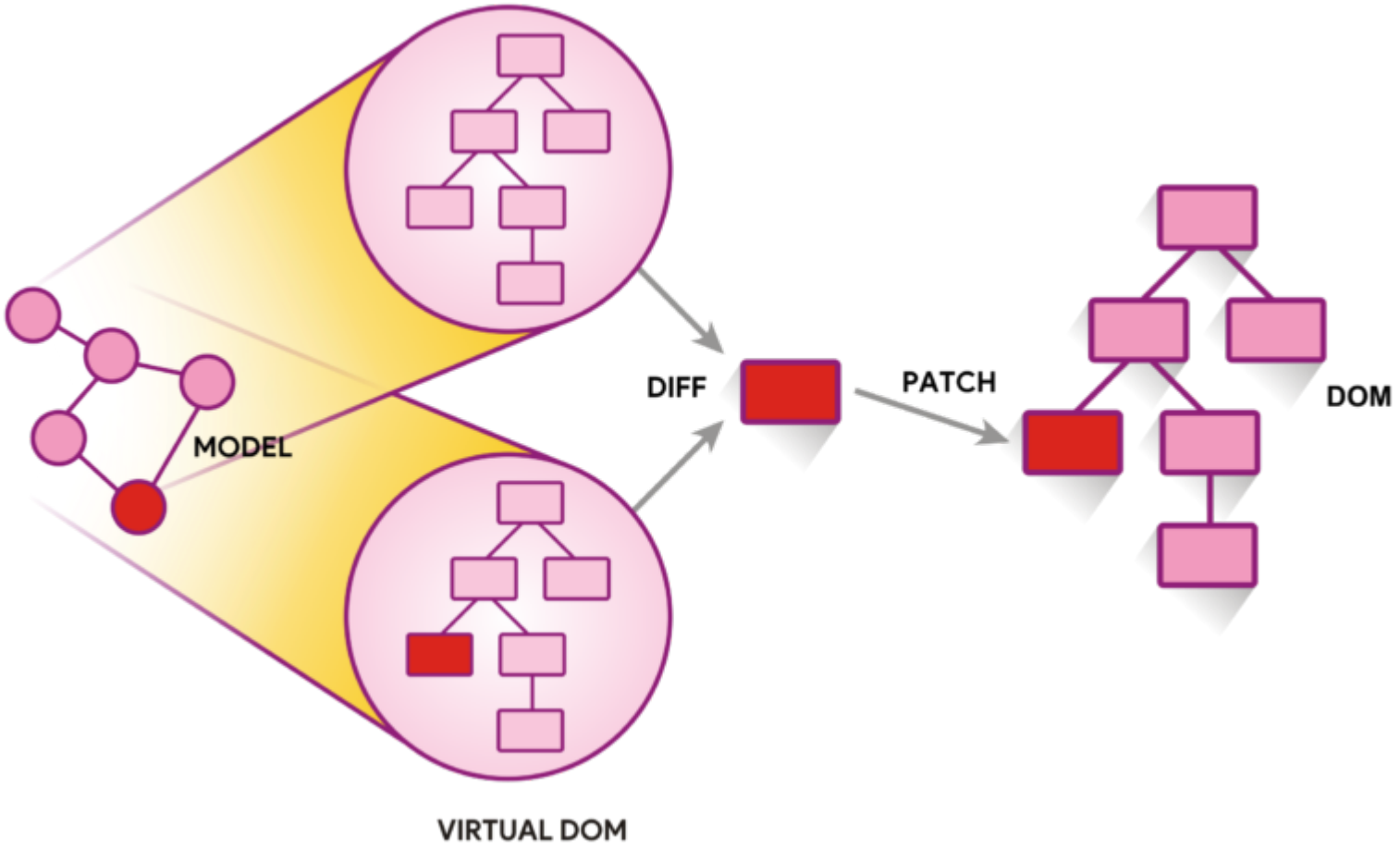


Схема из http://www.systango.com/blog/react-js-infinity/

Когда использовать C# + WebAssembly

- Если вы остаётесь внутри экосистемы «заточенного» под неё фреймворка
- Вам не нужно плотное взаимодействие с JS
- Вы пытаетесь перенести в браузер имеющийся не-UI код

Bridge.NET

```
public static void Main()
{
  var div = Document.CreateElement("div");
  Document.Body.AppendChild(div);

  var textBox = Document.CreateElement<HTMLInputElement>("input");
  textBox.Value = "123321";
  div.AppendChild(textBox);
  var button = Document.CreateElement<HTMLButtonElement>("button");
  button.AppendChild(Document.CreateTextNode("Click me!"));
  div.AppendChild(button);

  button.OnClick = ev =>
  {
    Window.Alert($"{textBox.Value}\nx: {ev.PageX}, y: {ev.PageY}");
  };
}
```


ДЕМО

```
Bridge.assembly("SimpleBridge", function ($asm, globals) {
  "use strict";
  Bridge.define("SimpleBridge.Program", {
    main: function Main () {
      var div = document.createElement("div");
      document.body.appendChild(div);
      var textBox = document.createElement("input");
      textBox.value = "123321";
      div.appendChild(textBox);
      var button = document.createElement("button");
      button.appendChild(document.createTextNode("Click me!"));
      div.appendChild(button);
      button.onclick = function (ev) {
        window.alert(System.String.format("{0}\nx: {1}, y: {2}",
textBox.value, Bridge.box(ev.pageX, System.Int32), Bridge.box(ev.pageY,
System.Int32)));
      };
    }
  });
});
```


```
Bridge.assembly("SimpleBridge", function ($asm, globals) {
  "use strict";
  Bridge.define("SimpleBridge.Program", {
    main: function Main () {
      var div = document.createElement("div");
      document.body.appendChild(div);
      var textBox = document.createElement("input");
      textBox.value = "123321";
      div.appendChild(textBox);
      var button = document.createElement("button");
      button.appendChild(document.createTextNode("Click me!"));
      div.appendChild(button);
      button.onclick = function (ev) {
        window.alert(System.String.format("{0}\nx: {1}, y: {2}",
textBox.value, Bridge.box(ev.pageX, System.Int32), Bridge.box(ev.pageY,
System.Int32)));
      };
    }
  });
});
```


```
Bridge.assembly("SimpleBridge", function ($asm, globals) {
  "use strict";
  Bridge.define("SimpleBridge.Program", {
    main: function Main () {
      var div = document.createElement("div");
      document.body.appendChild(div);
      var textBox = document.createElement("input");
      textBox.value = "123321";
      div.appendChild(textBox);
      var button = document.createElement("button");
      button.appendChild(document.createTextNode("Click me!"));
      div.appendChild(button);
      button.onclick = function (ev) {
        window.alert(System.String.format("{0}\nx: {1}, y: {2}",
textBox.value, Bridge.box(ev.pageX, System.Int32), Bridge.box(ev.pageY,
System.Int32)));
      };
    }
  });
});
```

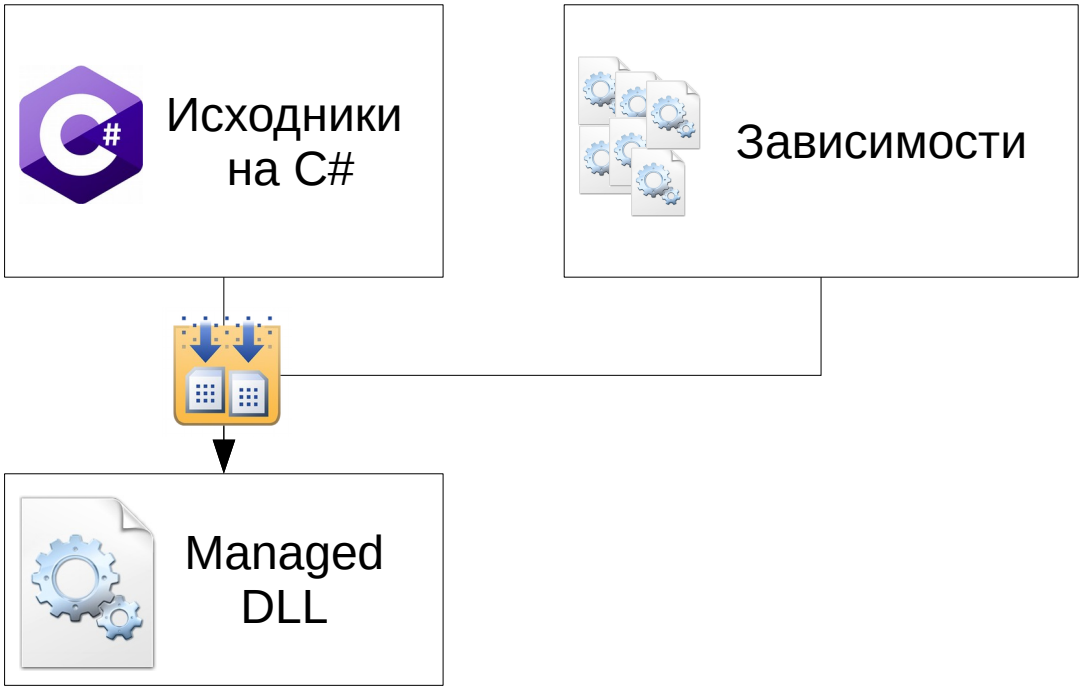

```
return {
  $boxed: true,
  fn: {
    toString: toStr,
    getHashCode: hashCode
  },
  v: v,
  type: T,
  constructor: T,
  getHashCode: function() {
    return this.fn.getHashCode ? this.fn.getHashCode(this.v) : Bridge.getHashCode(this.v);
  },
  equals: function (o) {
    if (this === o) {
      return true;
    }
    var eq = this.equals;
    this.equals = null;
    var r = Bridge.equals(this.v, o);
    this.equals = eq;
    return r;
  },
  valueOf: function() {
    return this.v;
  },
  toString: function () {
    return this.fn.toString ? this.fn.toString(this.v) : this.v.toString();
  }
};
```

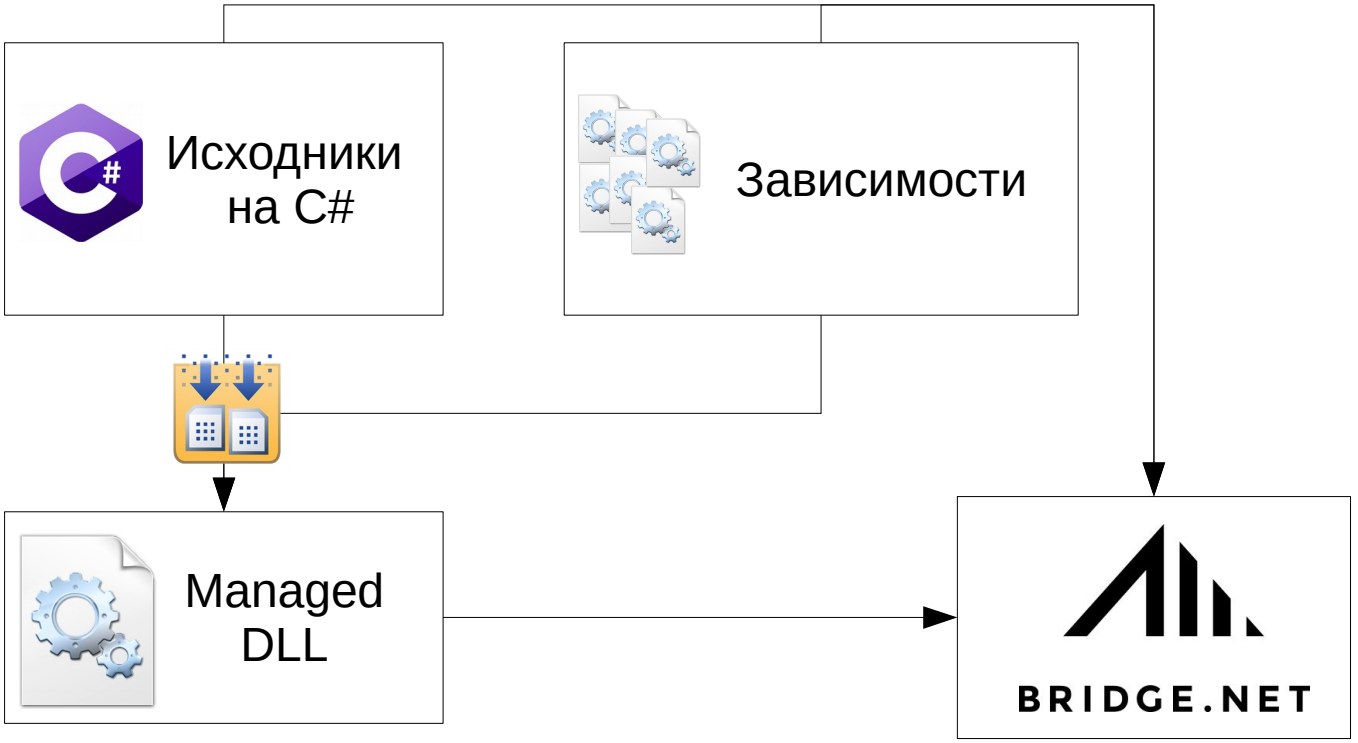

```
[Reflectable]
[Constructor("String")]
[External]
public sealed class String : IEnumerable, IBridgeClass, ICloneable, IEnumerable<char>,
IEquatable<string>
{
 [InlineConst]
 public const string Empty = "";

 [Convention(Notation.LowerCamelCase)]
 public extern int Length { get; }

 [Template("System.String.fromCharArray({value})")]
 public extern String(char[] value);

 [Template("System.String.concat({str0}, {str1})")]
 public static extern string Concat(string str0, string str1);

 [Template("System.String.fromCharCount({c}, {count})")]
 public extern String(char c, int count);

 [Template("System.String.fromCharArray({value}, {startIndex}, {length})")]
 public extern String(char[] value, int startIndex, int length);

 [Template("System.String.isNullOrEmpty({value})")]
```

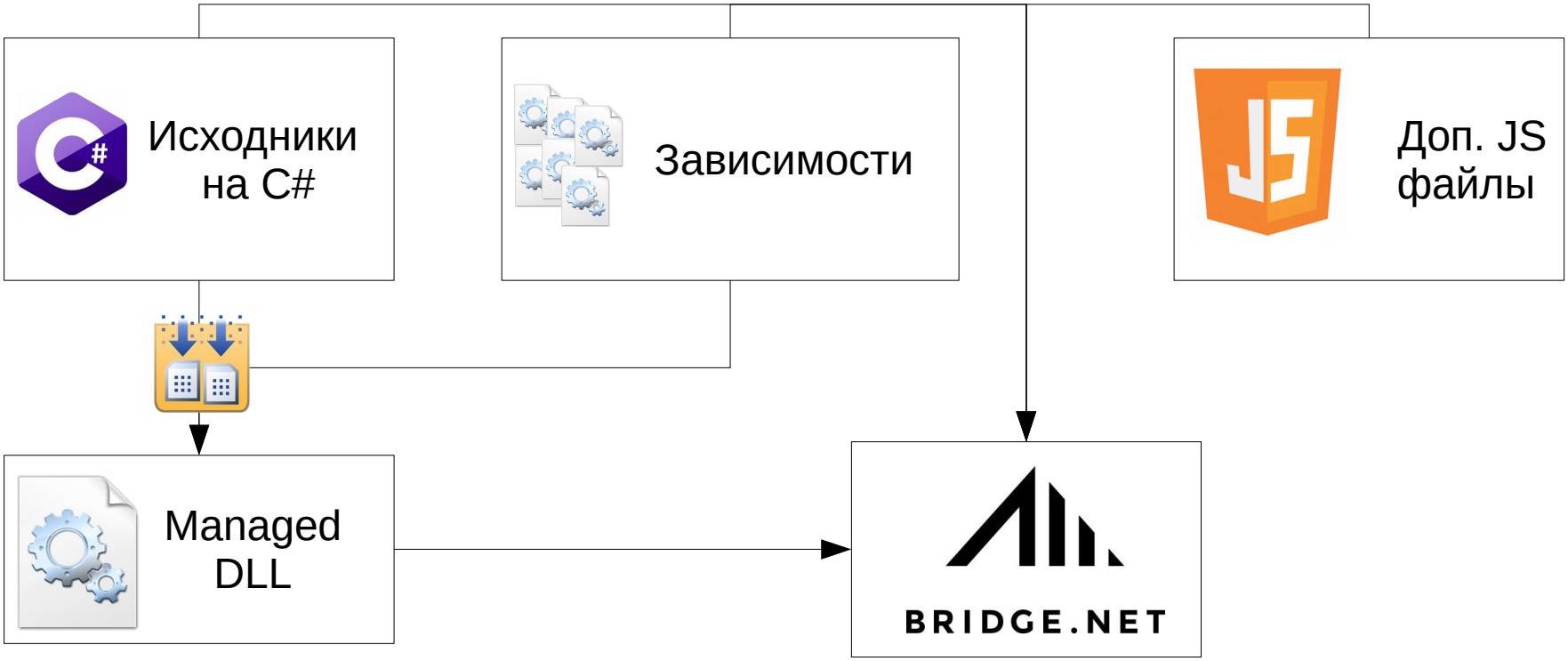

```
[Reflectable]
[Constructor("String")]
[External]
public sealed class String : IEnumerable, IBridgeClass, ICloneable, IEnumerable<char>,
IEquatable<string>
{
 [InlineConst]
 public const string Empty = "";

 [Convention(Notation.LowerCamelCase)]
 public extern int Length { get; }

 [Template("System.String.fromCharArray({value})")]
 public extern String(char[] value);

 [Template("System.String.concat({str0}, {str1})")]
 public static extern string Concat(string str0, string str1);

 [Template("System.String.fromCharCount({c}, {count})")]
 public extern String(char c, int count);

 [Template("System.String.fromCharArray({value}, {startIndex}, {length})")]
 public extern String(char[] value, int startIndex, int length);

 [Template("System.String.isNullOrEmpty({value})")]
```

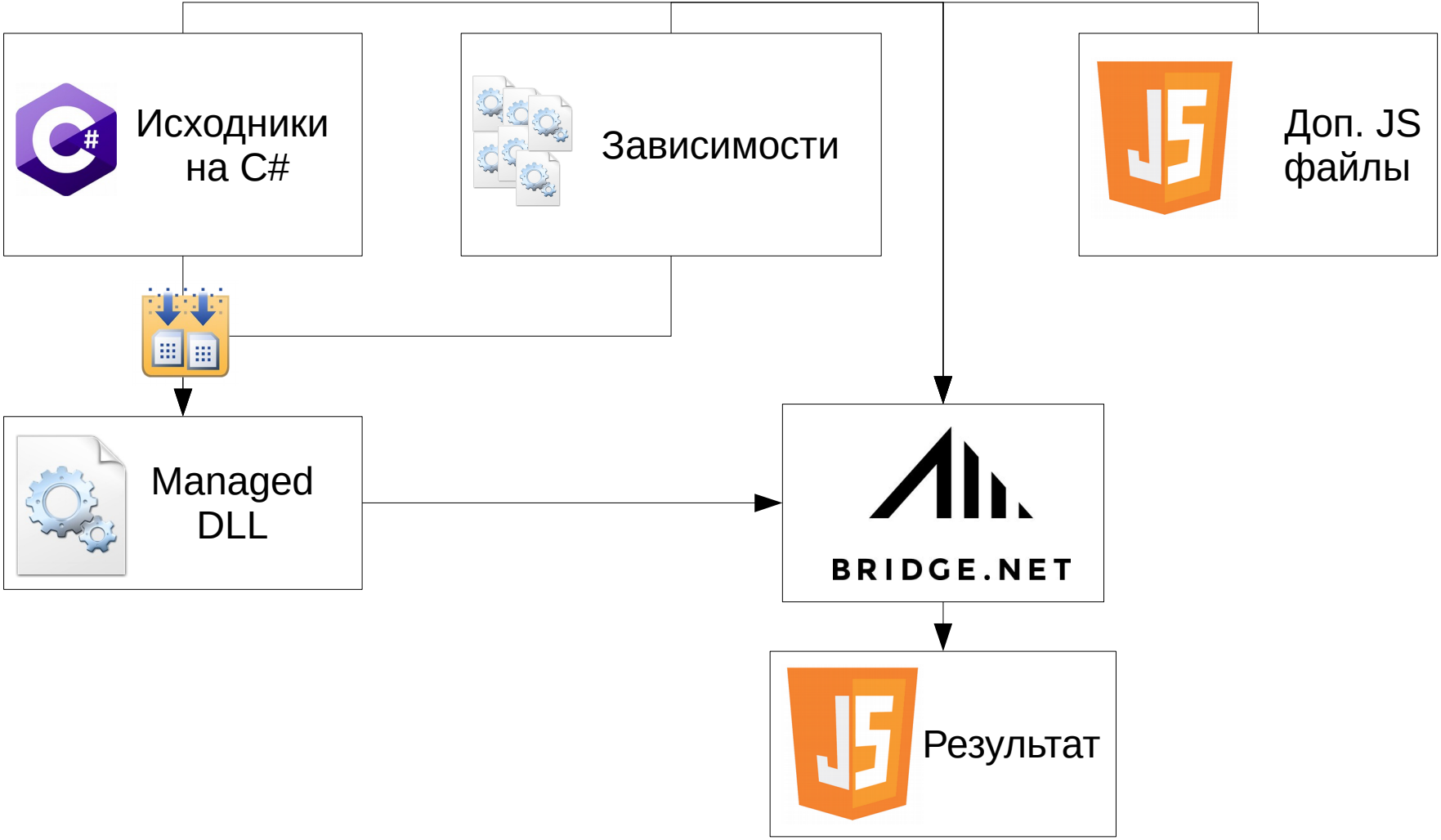

```
[Reflectable]
[Constructor("String")]
[External]
public sealed class String : IEnumerable, IBridgeClass, ICloneable, IEnumerable<char>,
IEquatable<string>
{
 [InlineConst]
 public const string Empty = "";

 [Convention(Notation.LowerCamelCase)]
 public extern int Length { get; }

 [Template("System.String.fromCharArray({value})")]
 public extern String(char[] value);

 [Template("System.String.concat({str0}, {str1})")]
 public static extern string Concat(string str0, string str1);

 [Template("System.String.fromCharCount({c}, {count})")]
 public extern String(char c, int count);

 [Template("System.String.fromCharArray({value}, {startIndex}, {length})")]
 public extern String(char[] value, int startIndex, int length);

 [Template("System.String.isNullOrEmpty({value})")]
```

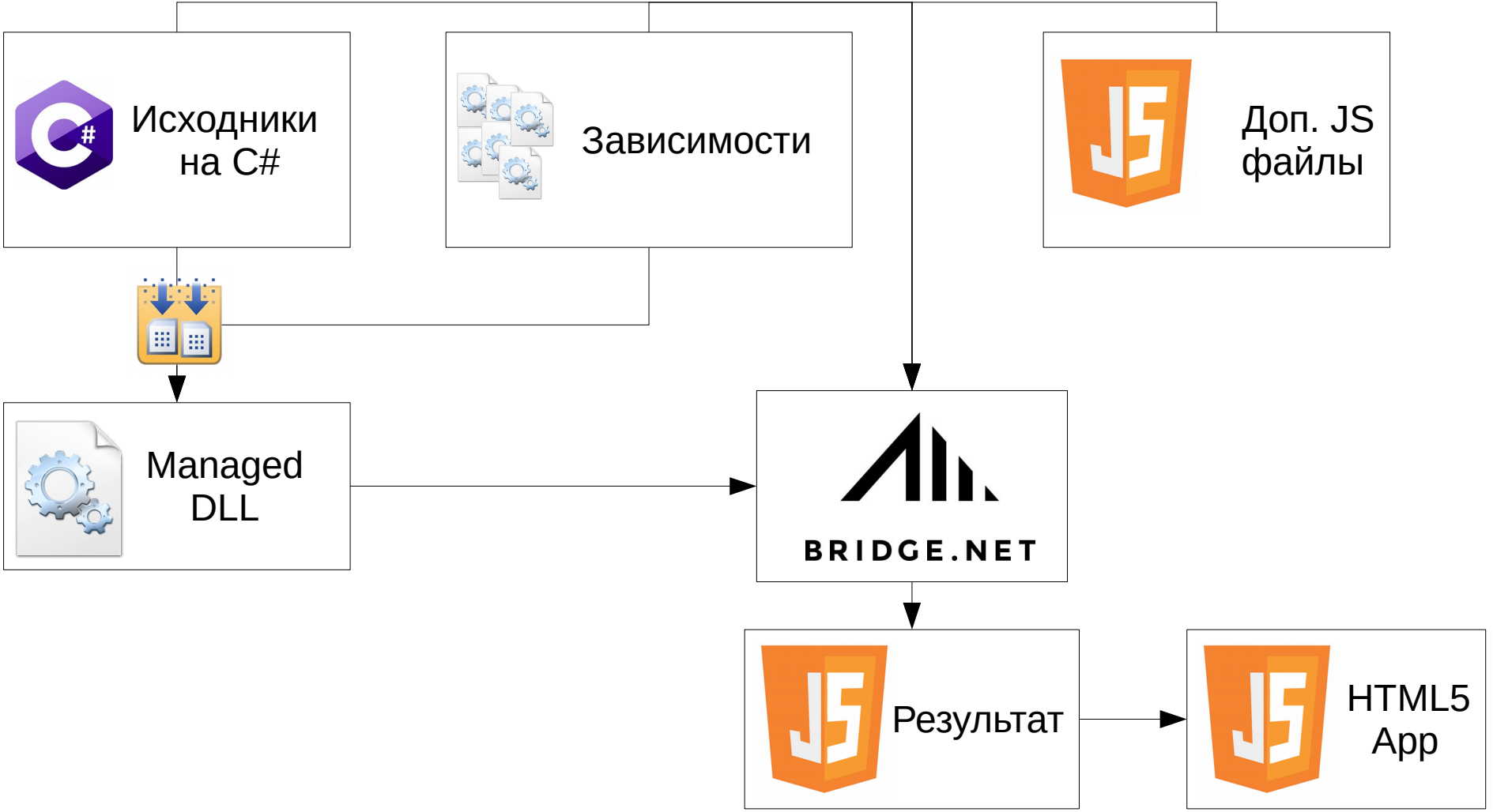

```
concat: function (values) {
  var list = (arguments.length == 1 &&
              Array.isArray(values))
                  ? values
                  : [].slice.call(arguments),
    s = "";
  for (var i = 0; i < list.length; i++) {
    s += list[i] == null ? "" : Bridge.toString(list[i]);
  }
  return s;
},
```

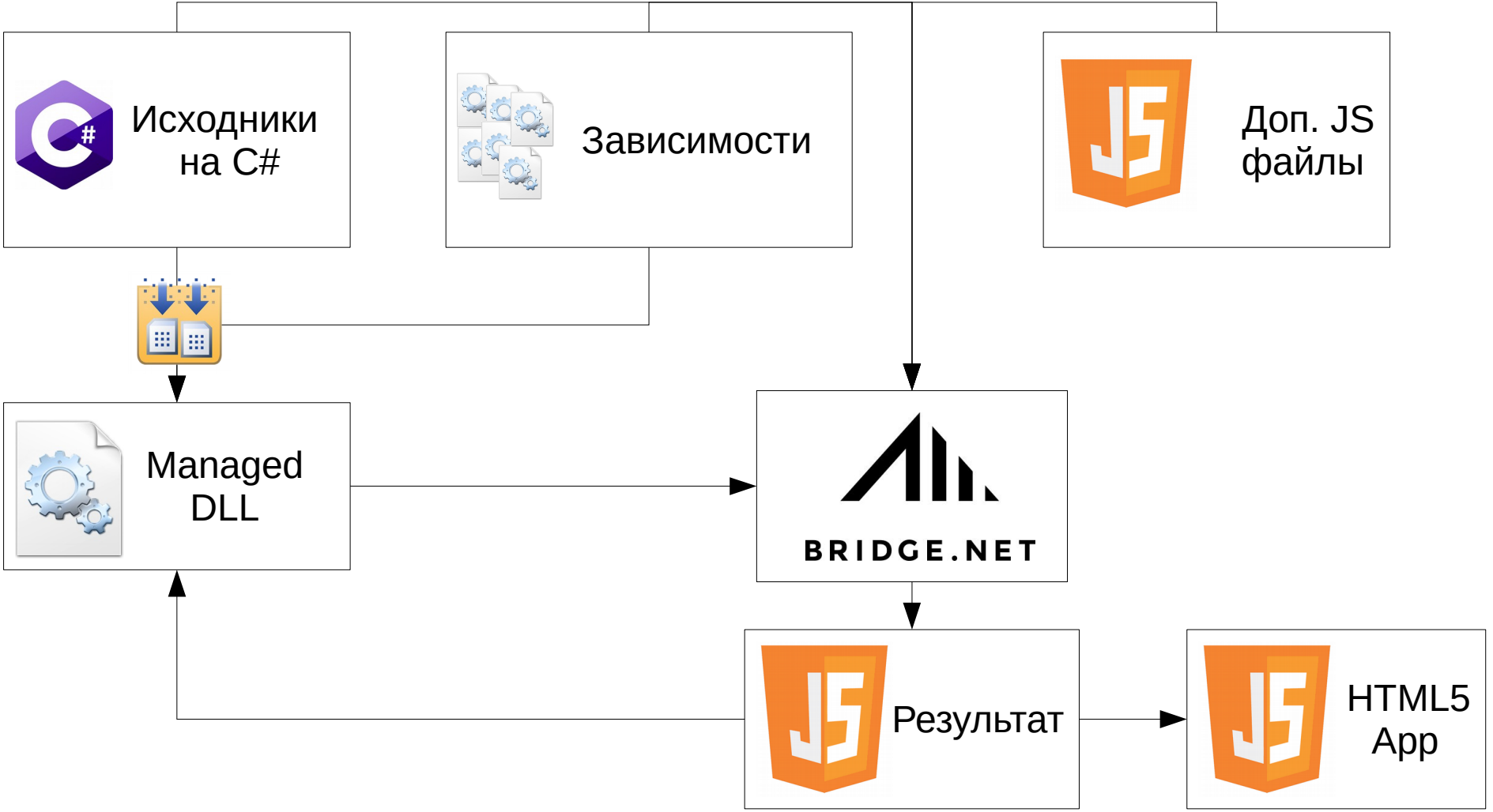


Исходники
на C#
Зависимости
Managed
DLL


Исходники
на C#
Зависимости
Managed
DLL
BRIDGE.NET


Исходники на C#
Зависимости
Доп. JS файлы
Managed DLL
BRIDGE.NET

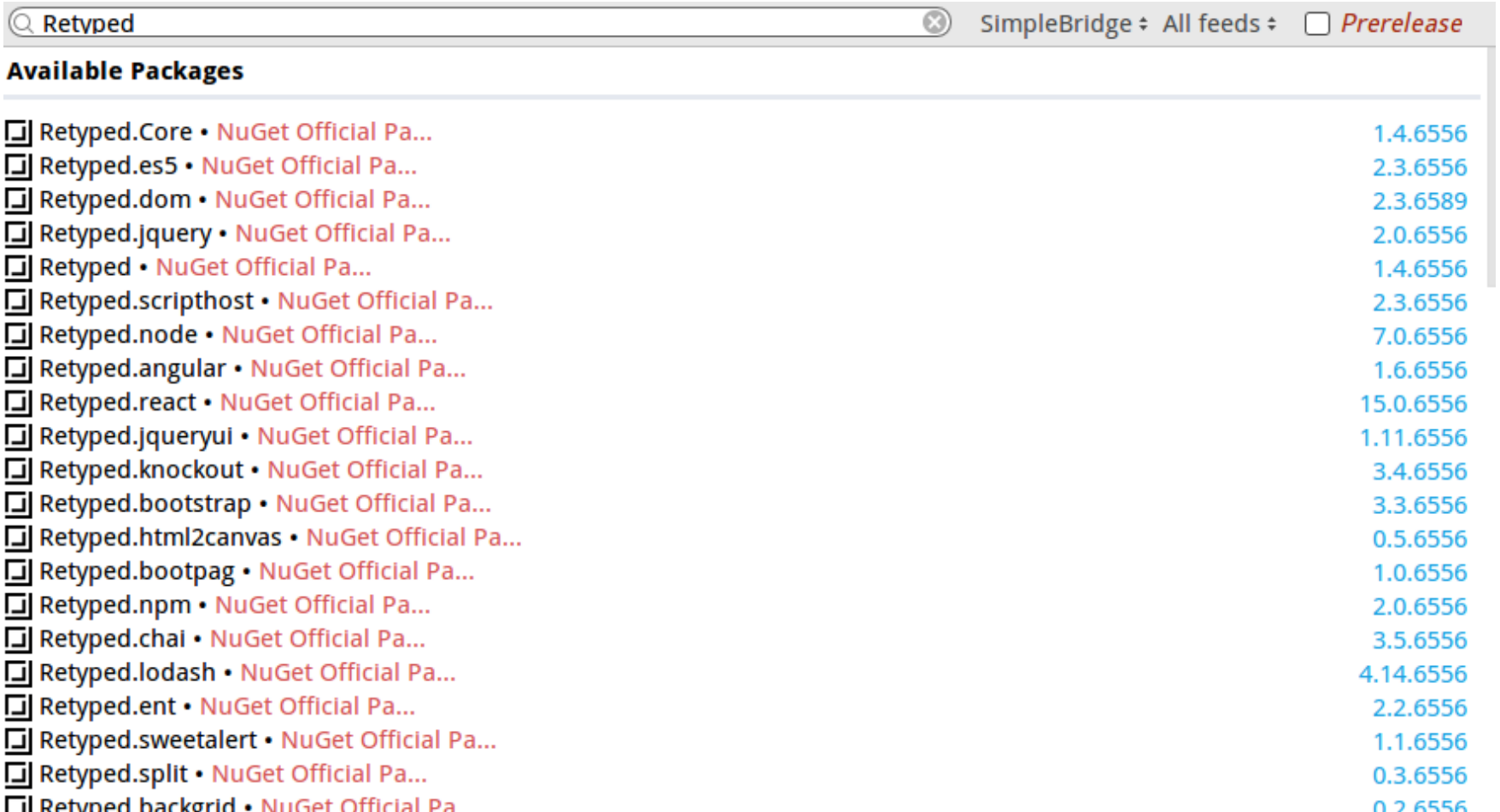

Исходники на C#
Зависимости
Доп. JS файлы
Managed DLL
BRIDGE.NET
Результат


Исходники на C#
Зависимости
Доп. JS файлы
Managed DLL
BRIDGE.NET
Результат
HTML5 App

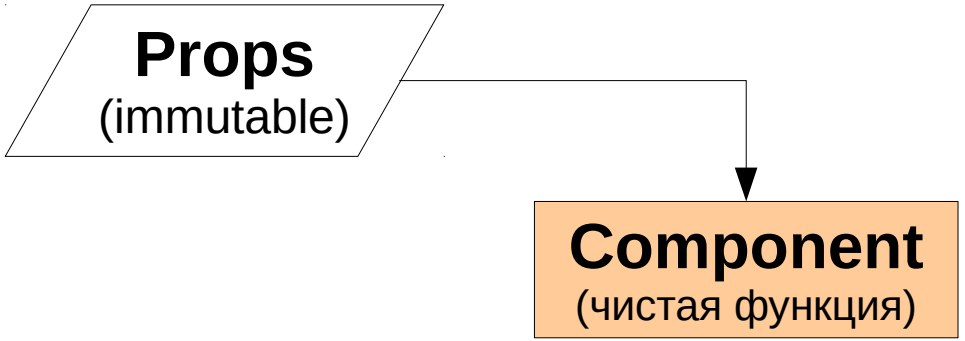

Исходники на C#
Зависимости
Доп. JS файлы
Managed DLL
BRIDGE.NET
Результат
HTML5 App

Retyped - генерация обёрток по .d.ts

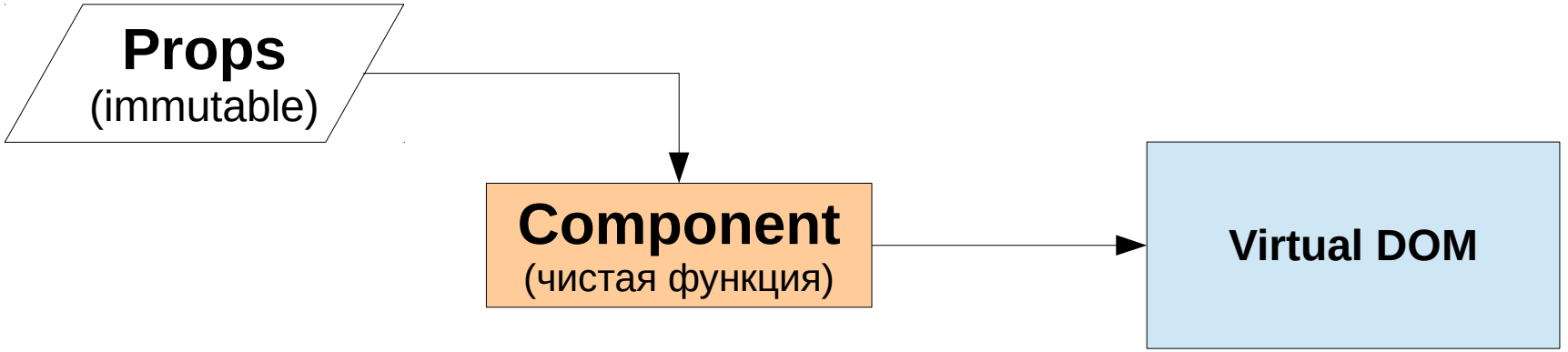
2300+ готовых к использованию JS-библиотек



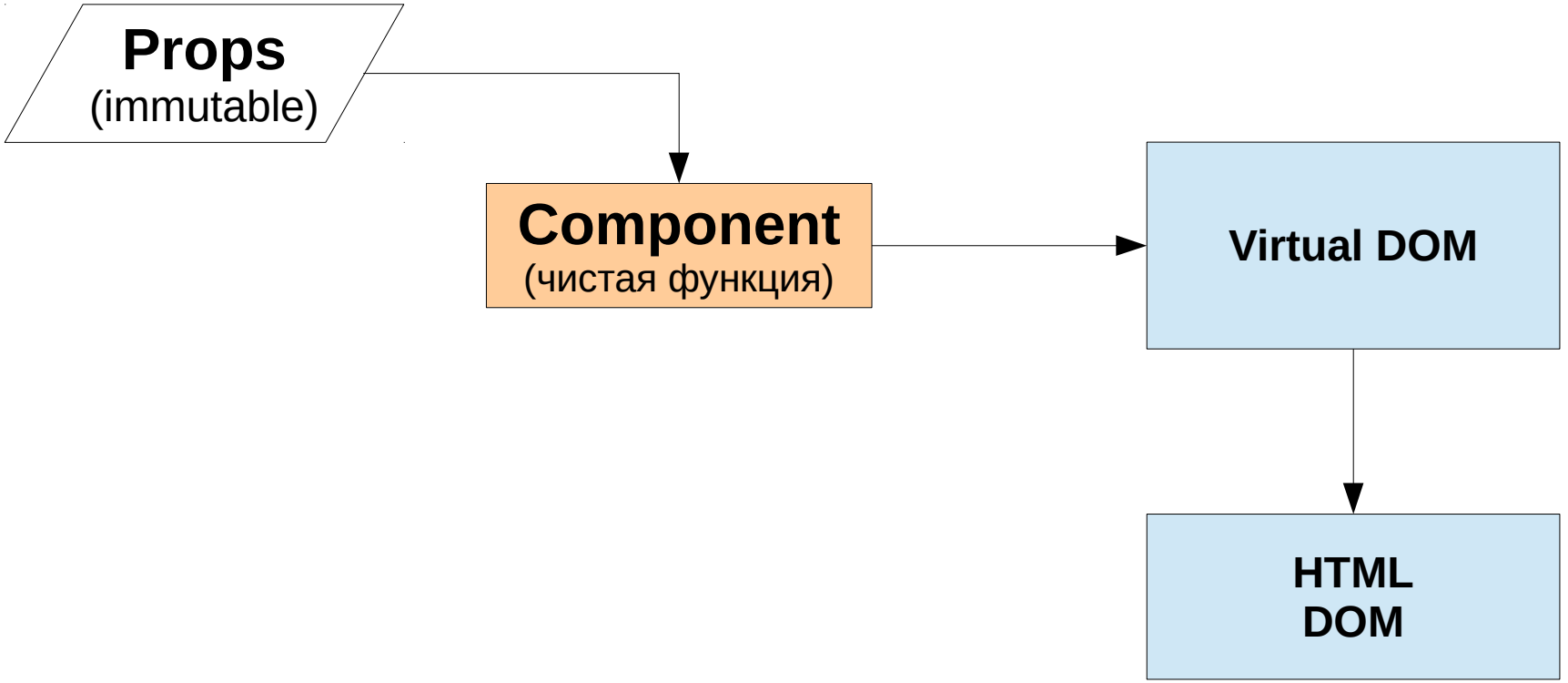
React



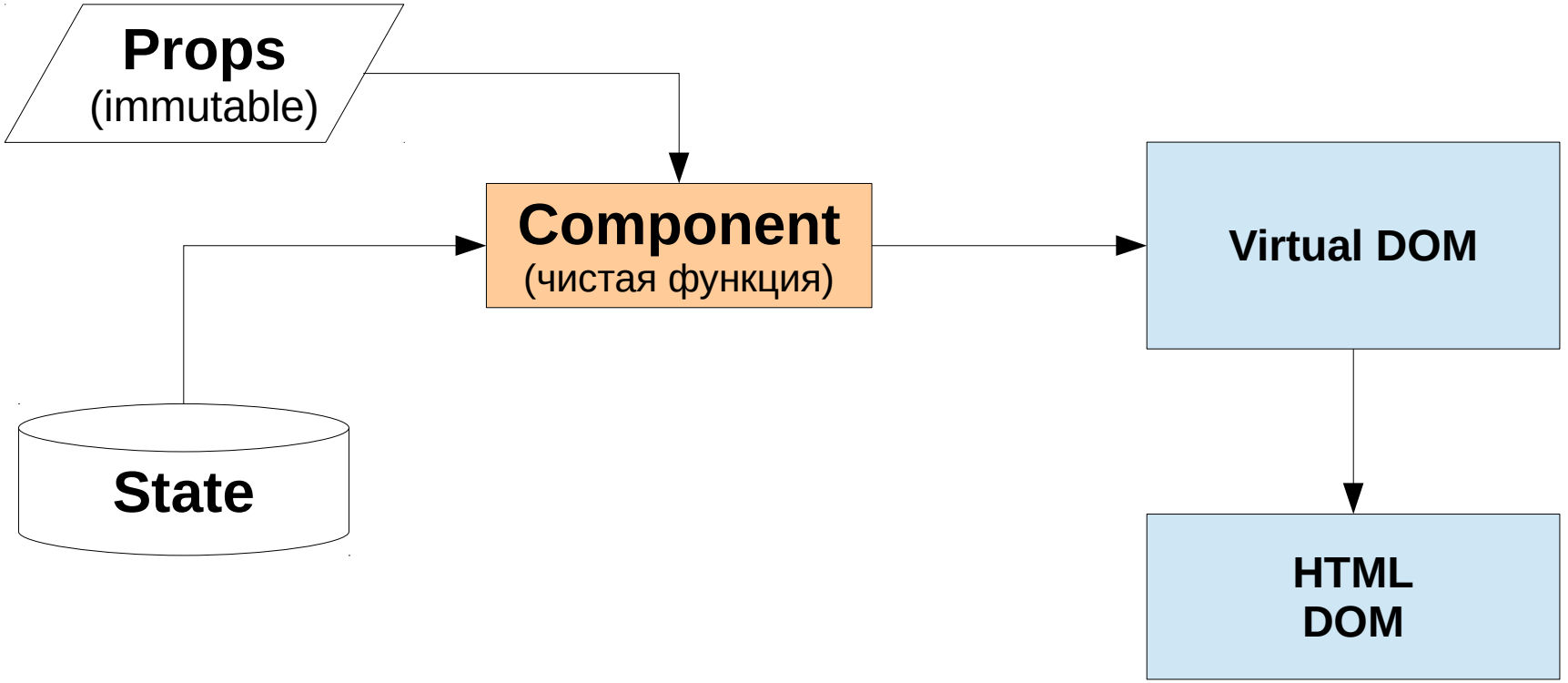
React



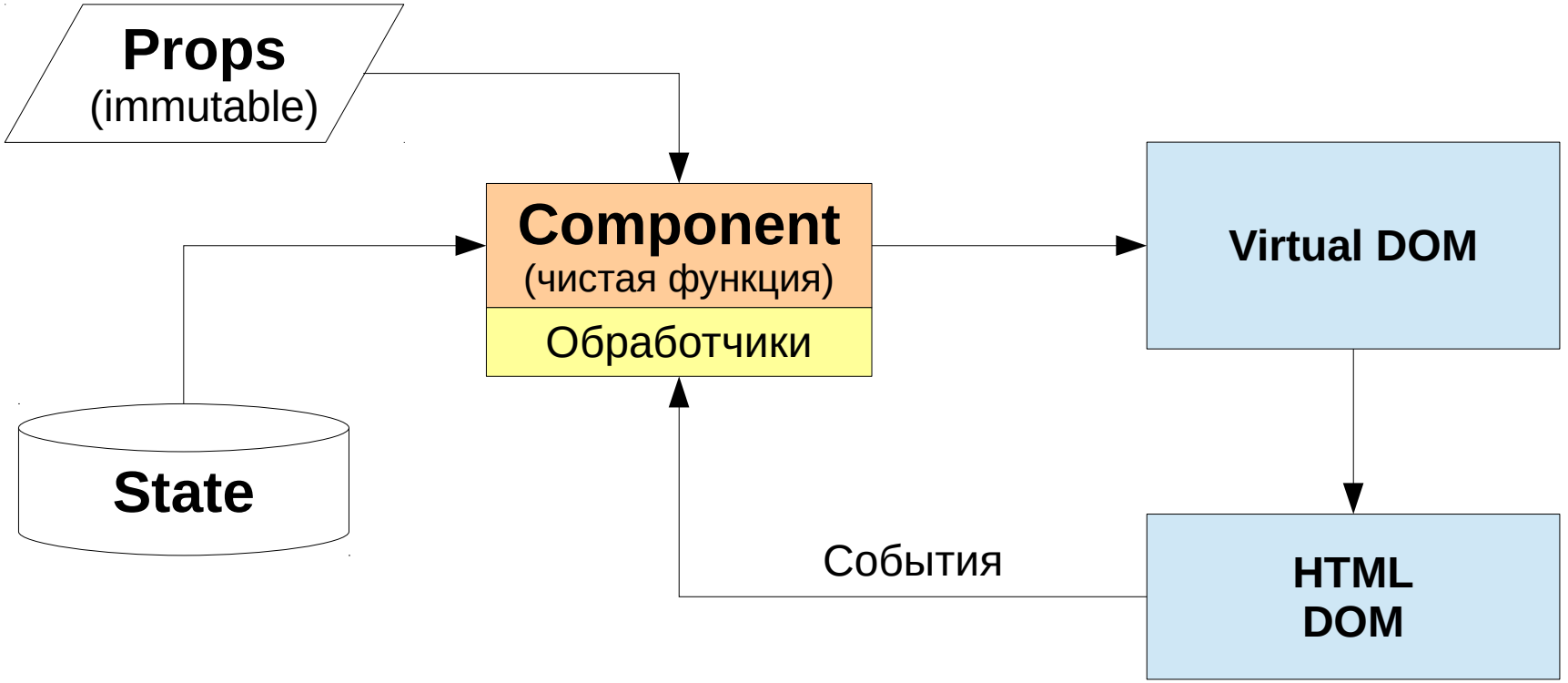
React


Props
(immutable)
Component
(чистая функция)
Virtual DOM
HTML
DOM

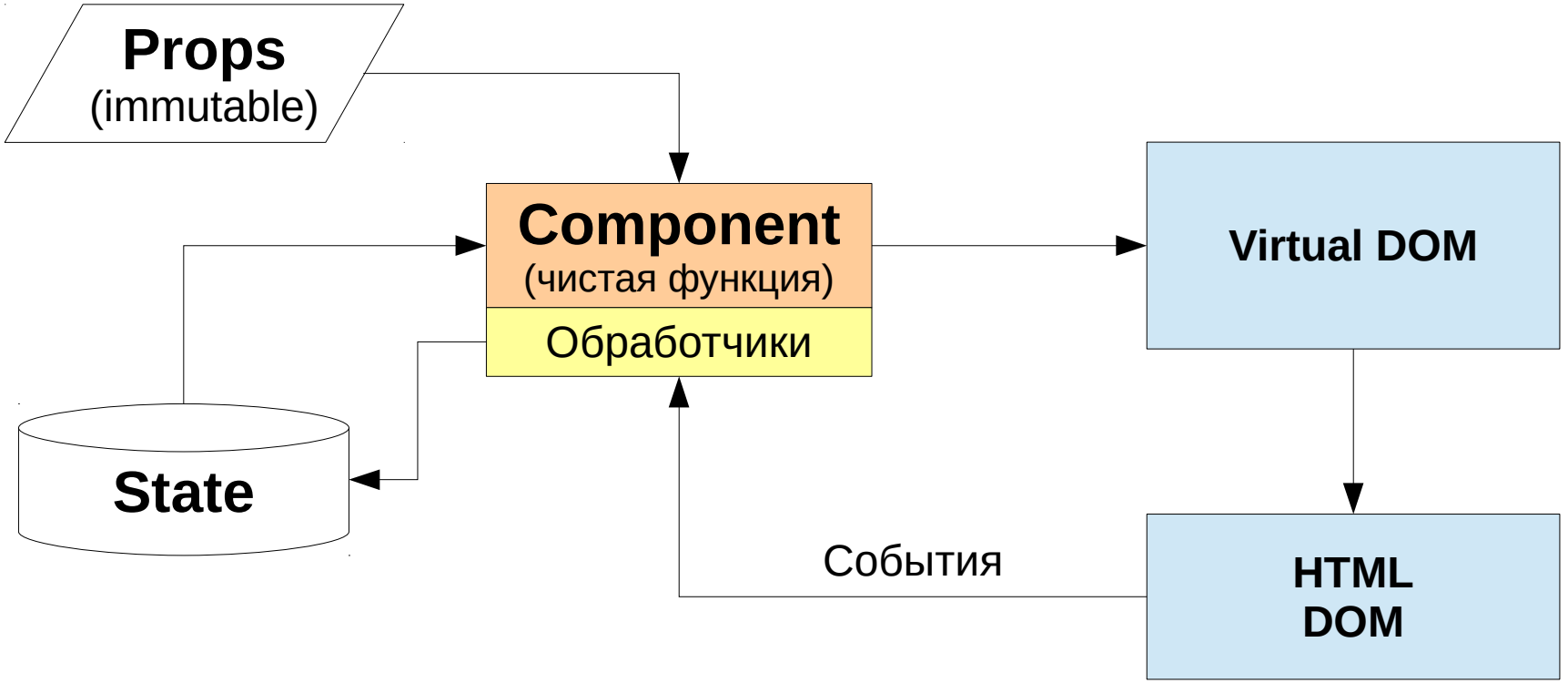
React


Props
(immutable)
Component
(чистая функция)
Virtual DOM
State
HTML
DOM

React


Props
(immutable)
Component
(чистая функция)
Обработчики
State
Virtual DOM
HTML
DOM
События

React


Props
(immutable)
Component
(чистая функция)
Обработчики
State
Virtual DOM
HTML
DOM
События

ДЕМО

Преимущества и недостатки Bridge.NET

Преимущества и недостатки Bridge.NET

+ Бесшовная работа с JS и DOM
+ Удобная работа с имеющимися JS-библиотеками
+ Размер приложения

- Доступна не вся BCL
- Нет совместимости с .NET Standard
- Библиотеки необходимо компилировать под Bridge

Спикер

Никита Цуканов

nikita.d.tsukanov@gmail.com

Telegram: kekekeks