Nau Engine

Александр Зеленщиков

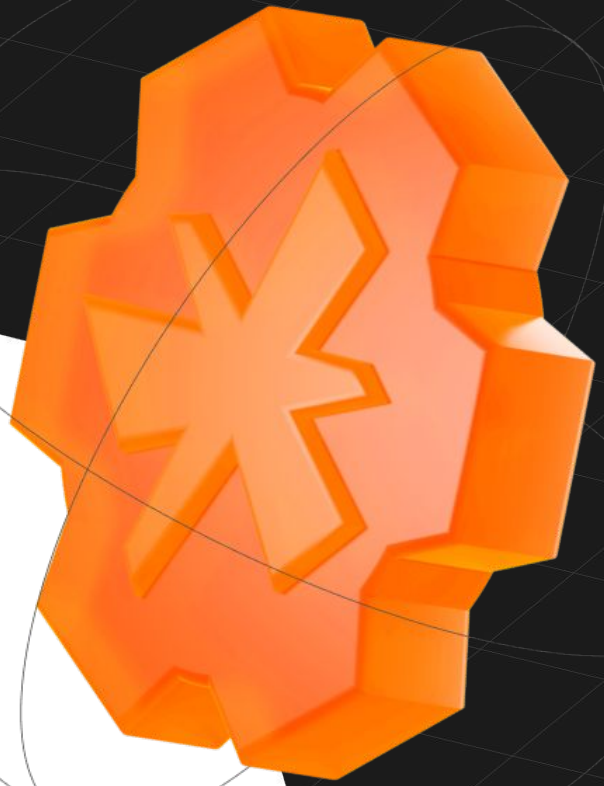
Агностическая скриптовая система для игрового движка Nau Engine

StreamTheater

Содержание

NauEngine | StreamTheater

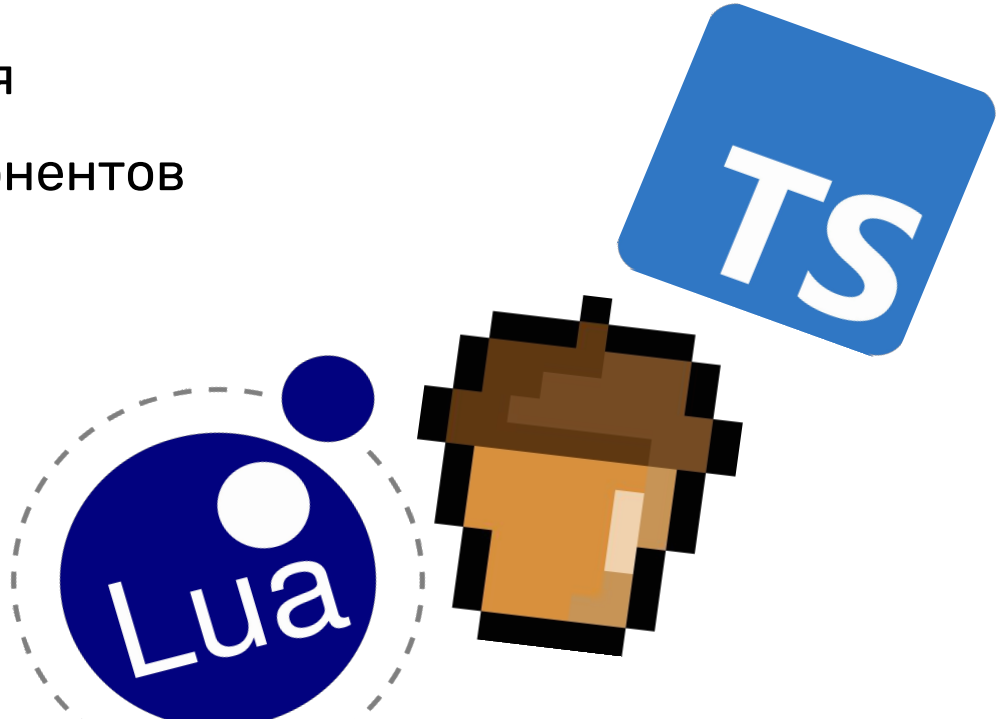
01 | Введение и мотивация



Что такое скриптинг?

NauEngine | StreamTheater

- Быстрая разработка
- Низкий порог вхождения
- Один из основных компонентов игрового движка



Зачем много языков в игровом движке?

NauEngine | StreamTheater

- Пиши код на чем привык
- Не выбирай что подходит, бери все что полезно

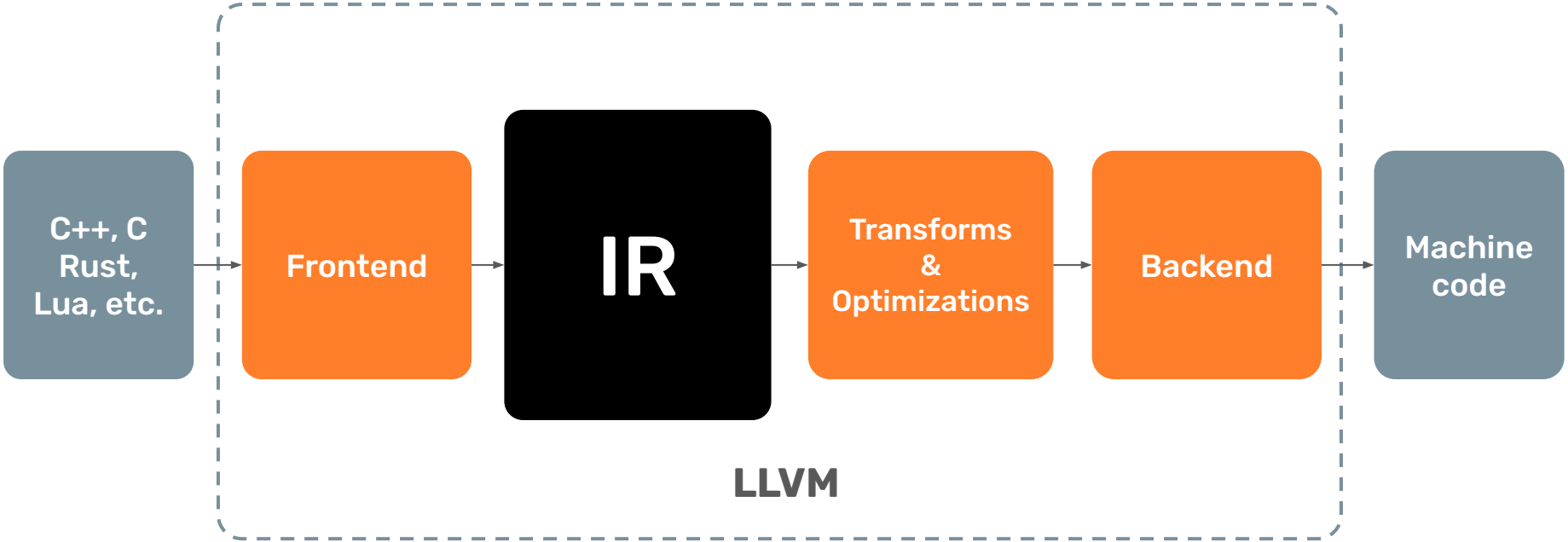
Что же делать?

- Интегрировать каждый язык по-своему
- Взять какую-нибудь платформу
  - .NET
  - JVM
  - Beem =) (Шутка, но мало ли)
- Может llvm, подумали мы?
  А почему бы и нет?

Что такое llvm?

NauEngine | StreamTheater

Это не виртуальная машина!



C++ → frontend → IR

NauEngine | StreamTheater

```
int foo(int a, int b)
{
  return a + b;
}
```

```
define i32 @foo(i32 %a, i32 %b)
{
  %result = add i32 %a, i32 %b
  ret i32 %result
}
```

Почему мы взяли llvm?

NauEngine | StreamTheater

- Нативная линковка скриптов с ядром движка
- Единая схема биндинга
- Широчайший спектр бэкендов
  - Все консоли
  - В том числе WASM
- Реализация JIT и AOT из коробки

02 | # Как это выглядит?

Как это выглядит?

NauEngine | StreamTheater

Lua

```lua
function foo(a, b)
  return a + b
end
function bar(a)
  return a * 100
end
```

C++

```cpp
struct MyScript
{
    virtual double foo(int a, int b) = 0;
    virtual int bar(double a) = 0;
}
```

C++ → Script

NauEngine | StreamTheater

```
DEFINE_SCRIPT_INTERFACE(MyScript,
struct MyScript
{
    virtual double foo(int a, int b) = 0;
    virtual void bar(double a) = 0;
};
)

    // ... где-то в коде ниже

auto module = core->newScriptModule<MyScript>("myscript.lua");
auto instance = module->newInstance();

instance->foo(10, 20);
instance->bar(1.0);
```

Script → C++

NauEngine | StreamTheater

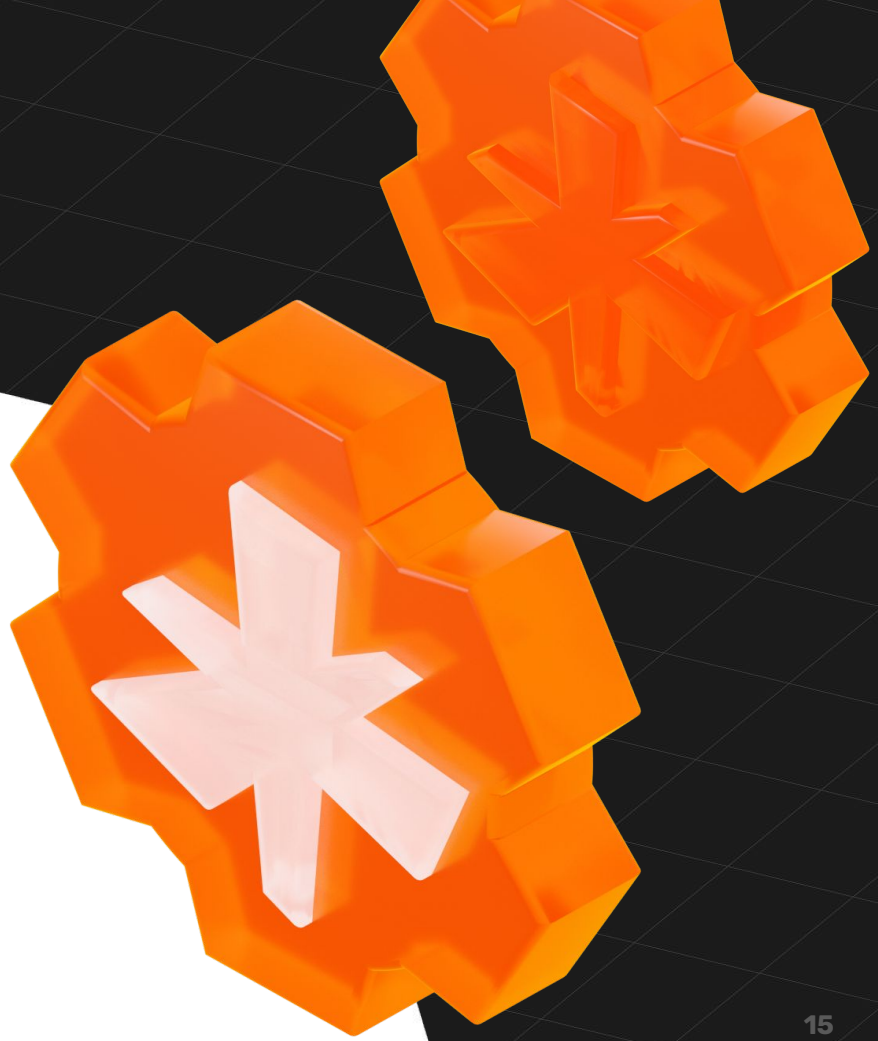
```
DEFINE_SCRIPT_INTERFACE(Logger,
struct Logger
{
  virtual void log(const char* msg) = 0;
};
)

struct LoggerImpl : Logger
{
  void log(const char* msg) override;
};

// ... где-то в коде ниже

LoggerImpl logger;
core->registerInstance<Logger>(&logger, "logger");
```

Вызов Native из скрипта

NauEngine | StreamTheater

```
function foo()
   -- код
   logger.log("Hello, world!")
   -- код
end
```

03 | Как это устроено?



Что под капотом newInstance?

NauEngine | StreamTheater

```
auto module = core->newScriptModule<MyScript>("myscript.lua");
auto instance = module->newInstance();
instance->foo(10, 20);


// ... где-то внутри Core в упрощенном виде

struct ScriptObject
{
    void* m_vtable;
    explicit ScriptObject(void* vtable): m_vtable(vtable) { }
};

void* ScriptModuleRuntime::newInstance() const
{
    return new ScriptObject(m_vtable);
}
```

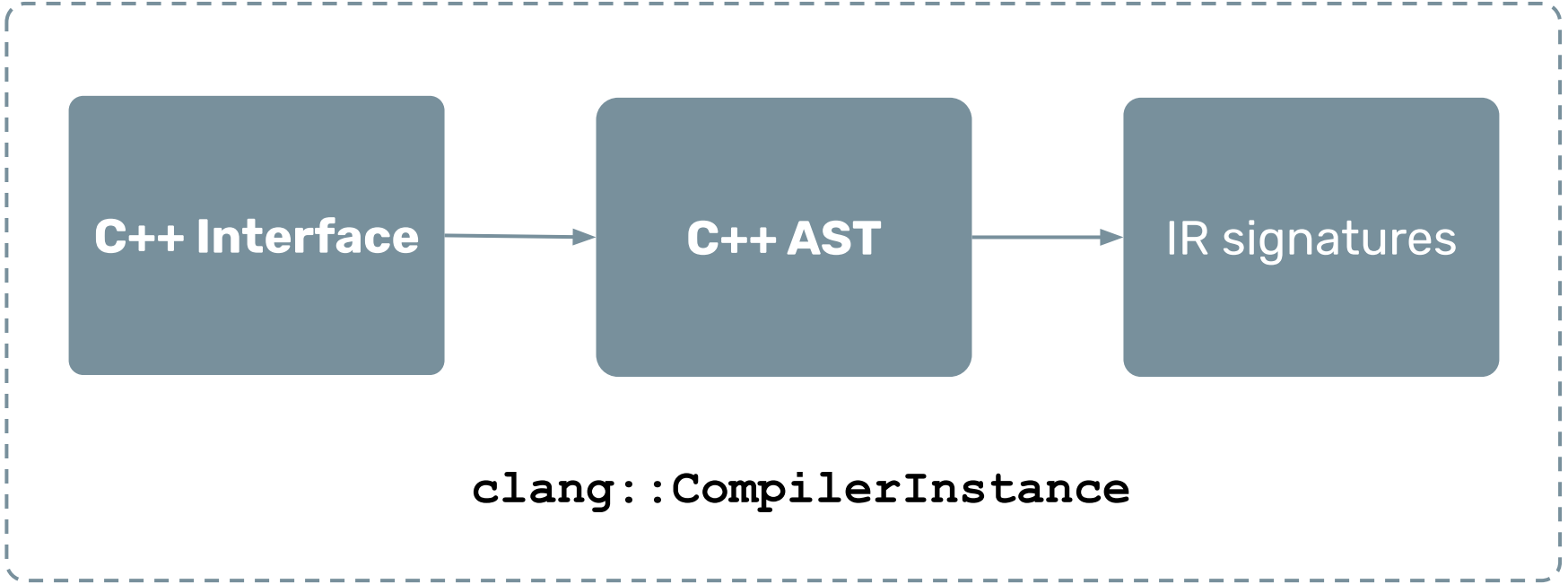
DEFINE_SCRIPT_INTERFACE

NauEngine | StreamTheater

```
template <typename T> const char* getSourceCode();
template <typename T> const char* getTypeName();

#define DEFINE_SCRIPT_INTERFACE(Type, I) \
I \
template <> const char* getSourceCode<Type>() { return #I; } \
template <> const char* getTypeName<Type>() { return #Type; } \
```

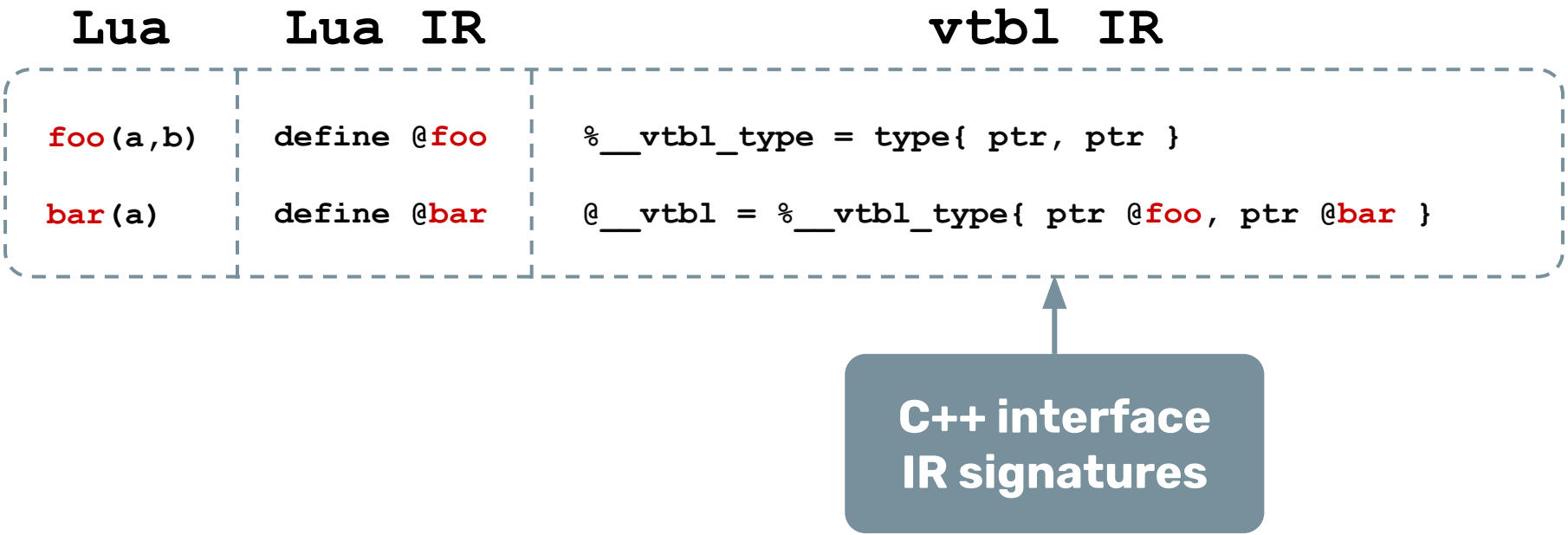
Что происходит с C++ интерфейсом?

NauEngine | StreamTheater



Как это работает? VTBL

NauEngine | StreamTheater



Режимы работы

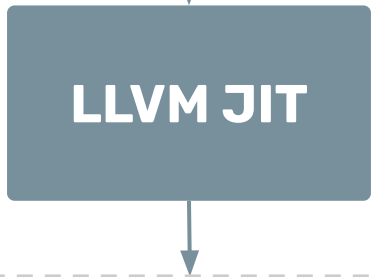
NauEngine | StreamTheater

- Dev Mode (JIT компиляция) // Игровой редактор
- Compile Mode (AOT компиляция) // Сборка игры
- Production Mode (JIT или AOT) // Игра у игроков =)

Работа в Dev-Mode

NauEngine | StreamTheater

```
define @foo     %__vtbl_type = type{ ptr, ptr }

define @bar     @__vtbl = %__vtbl_type{ ptr @foo, ptr @bar }
```




```
auto symbol = jit->lookup("__vtbl");
m_vtable = symbol.toPtr<void*>();
```

Линковка (AOT)

NauEngine | StreamTheater

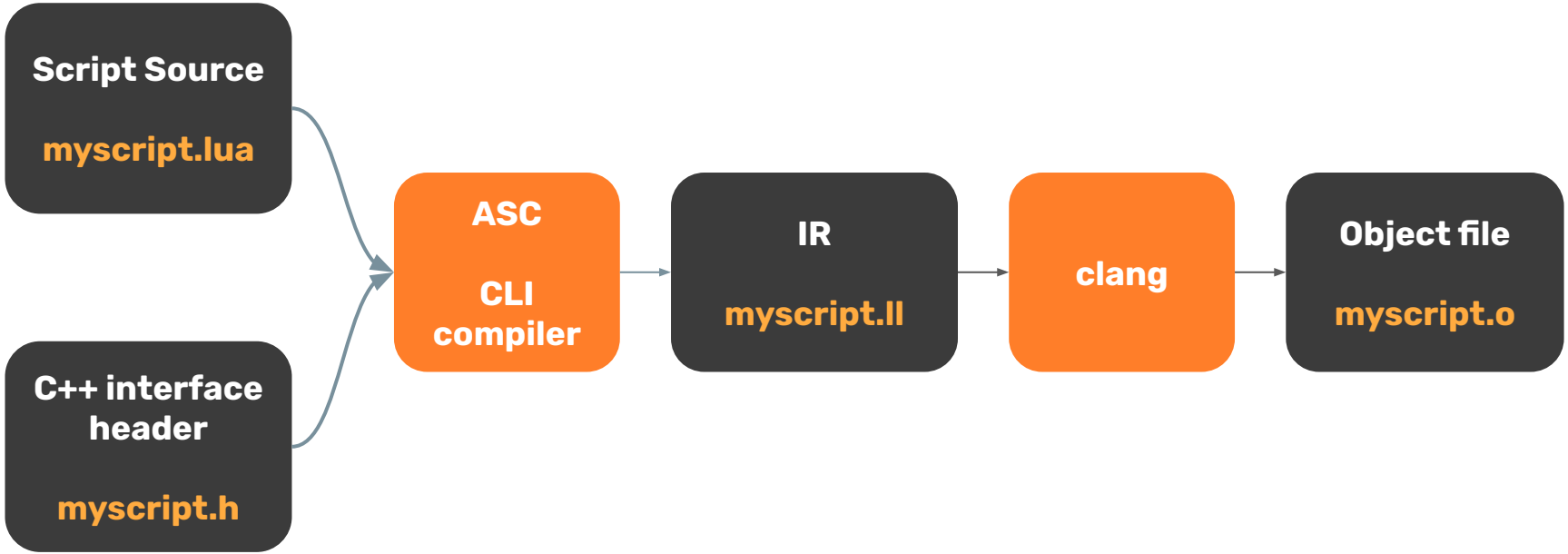
```
// Dev mode CMake
add_executable(Target main.cpp)
target_link_libraries(Target LuaIntegration)
```

```
// Prod mode CMake
add_executable(Target main.cpp)
target_link_scripts(Target scripts/myscript.lua)
target_link_libraries(Target LuaIntegration)
```

Компиляция (AOT)

// target_link_scripts

NauEngine | StreamTheater



Работа в Prod

NauEngine | StreamTheater

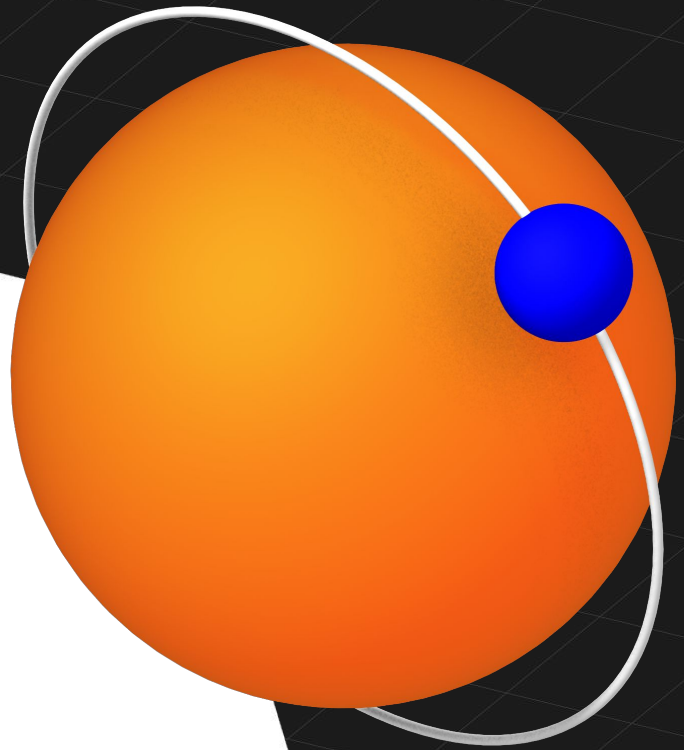
```
// AOT код скриптового модуля
@llvm.global_ctors = appending global ...

; ...
define internal void @.ctor_scripts_myscript() {
entry:
  call void @__asRegisterInit(ptr @.module_name, ptr @.vtable)
  ret void
}
```

Работа в Prod

NauEngine | StreamTheater

```
template<typename Interface>
std::shared_ptr<ScriptModule<Interface>> newScriptModule(const std::string& filename)
{
    const auto linked_module = getLinkedModule(filename);
    if (linked_module)
        return std::make_shared<ScriptModule<Interface>>(linked_module);

    const auto compiled_module = getCompiledModule(getInterface<Interface>(), filename);
    return std::make_shared<ScriptModule<Interface>>(compiled_module);
}
```

04 | Интеграция Lua



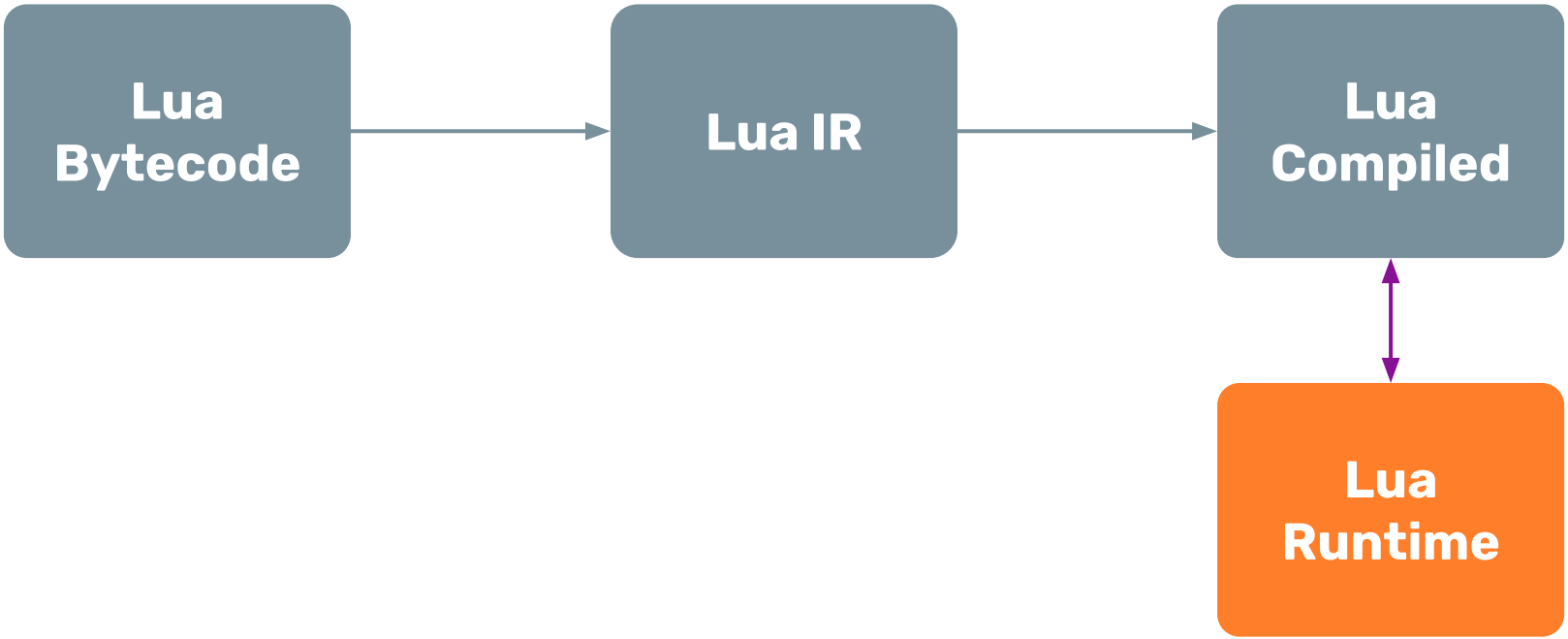
Языки или «О бедном рантайме замолвите слово...»

NauEngine | StreamTheater

- Статические языки
- Динамические языки
- .. и их рантаймы

Пример как встроен Lua

NauEngine | StreamTheater



Пример как встроен Lua

NauEngine | StreamTheater

// схлопывание стека

```
lua_getglobal(l, "foo");
lua_pushinteger(l, 10);
lua_pushinteger(l, 20);
lua_call(l, 2, 1);
```

call

```
//...
OP_ADD 1, 2
//...
```

```
define double @foo(double %a,double %b)
{
  ; ...
  %res = fadd double %a, %b
  ; ...
  ret double %res
}
```

Бенчмарки Lua

NauEngine | StreamTheater

Результаты замеров в миллисекундах:

title	native	lua as	lua 5.4.6	luajit 2.1
cycle	0.11	1.08	2.75	1.61
array	15.68	276.81	957.60	143.90
n queen	250.20	1585.12	5498.82	2280.69
life	1154.86	9804.57	63042.69	18377.10

Бенчмарки Lua

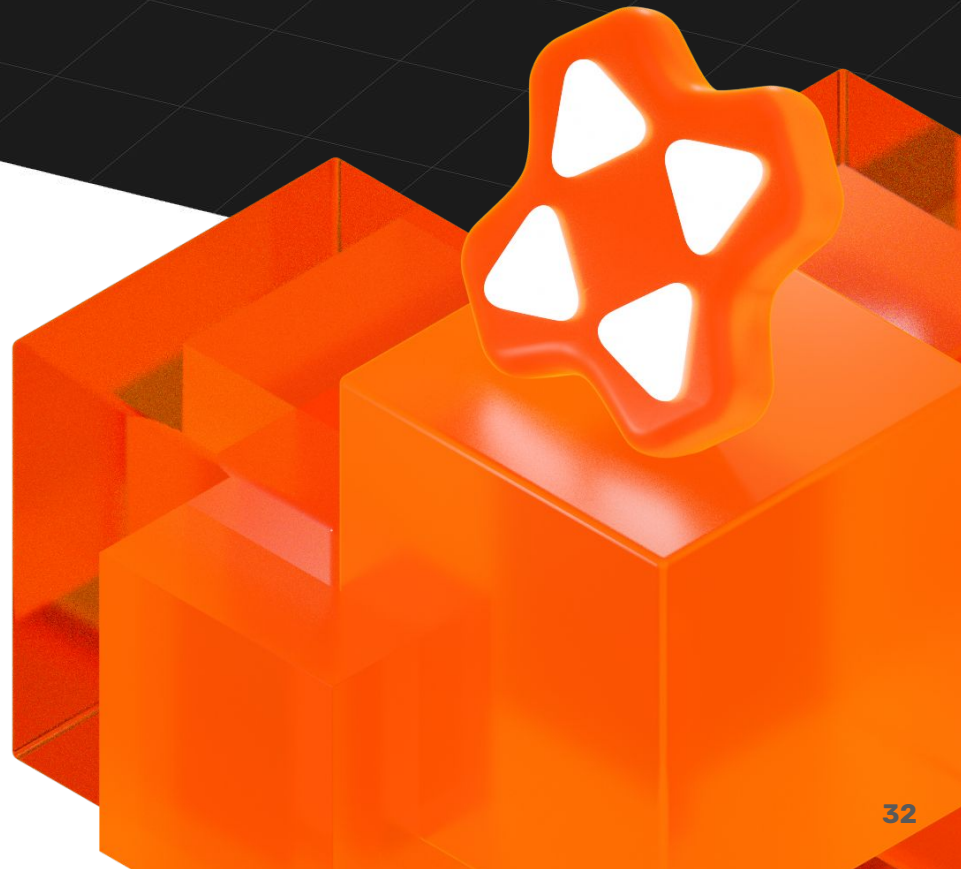
// кросс вызовы

NauEngine | StreamTheater

10 миллионов итераций на каждый тест.
Результаты замеров в миллисекундах:

title	lua as	lua 5.4.6	luajit 2.1
a + b	208.47	246.76	284.61
not a	159.68	233.12	238.77

05 | Выводы и планы



Планы по языкам

NauEngine | StreamTheater

- Lua *(подключен)*
- Squirrel *(подключен)*
- TypeScript *(в процессе)*
- C#
- daScript
- C++ как скриптовый язык (a la cling)

Плюсы

NauEngine | StreamTheater

- AOT для широкого спектра платформ
- Унифицированный биндинг
- .. в том числе между языками
- Производительность

Минусы

NauEngine | StreamTheater

- Есть llvm frontend и поехали? Не все так просто
- Подмножество языка
- Компиляция медленней интерпретации

Что дальше?

NauEngine | StreamTheater

- Gradual Typing от C++ интерфейса внутрь скрипта
- SSA для стекового рантайма
- Данные, а не только pure interfaces
- Hot reload кода в JIT Mode
- Отладка скриптов в JIT Mode

Вопросы?

Александр Зеленщиков

az@streamtheater.ru

@Sippul

Nau Engine

info@nauengine.org

@nauengine

Nau Engine | StreamTheater