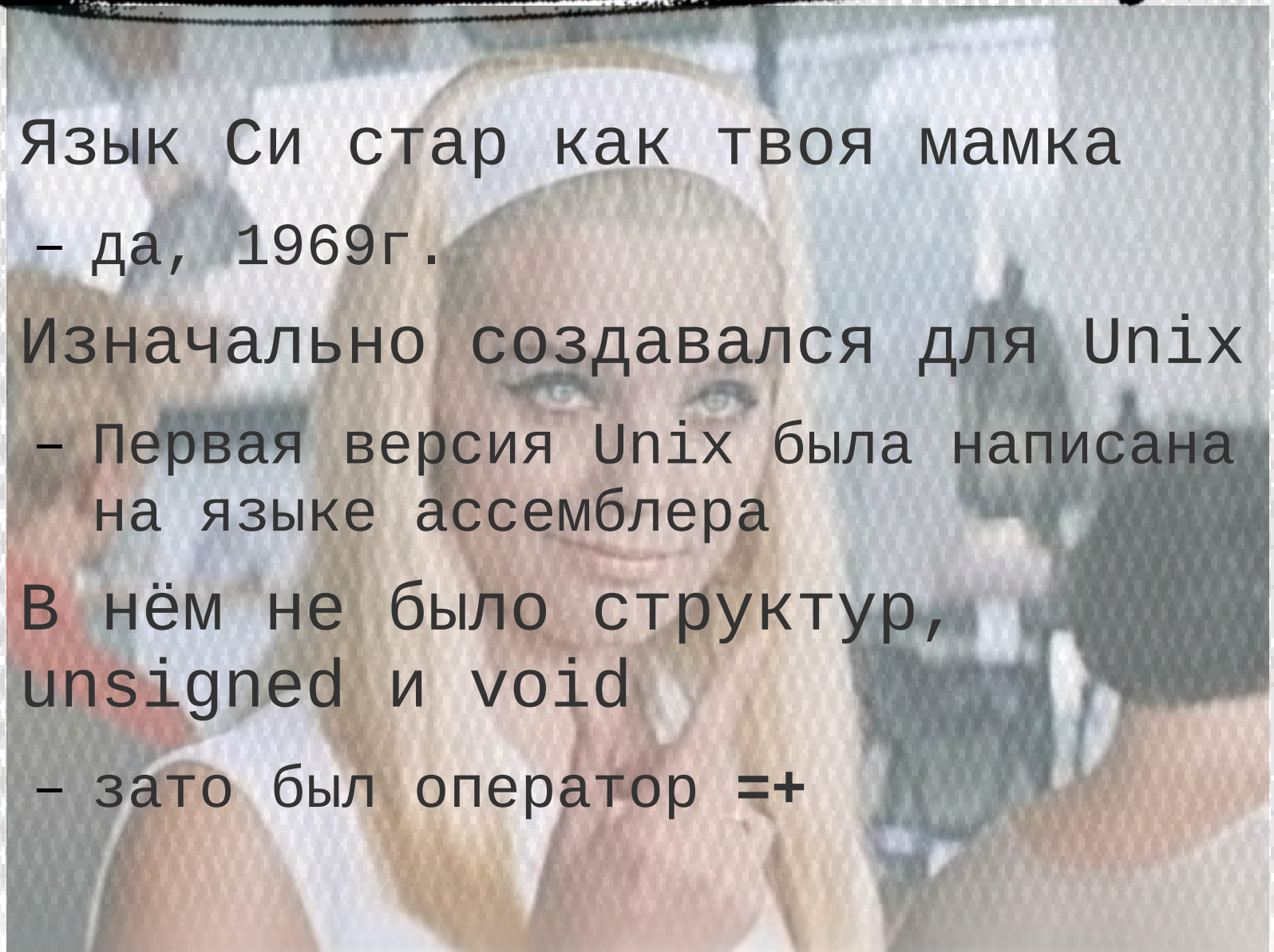
Между «до» и «ля»

Какие прелести сохранил

в современном мире?

Вадим Жуков <zhuk@openbsd.org>
LVEE 2018, Раков, Беларусь

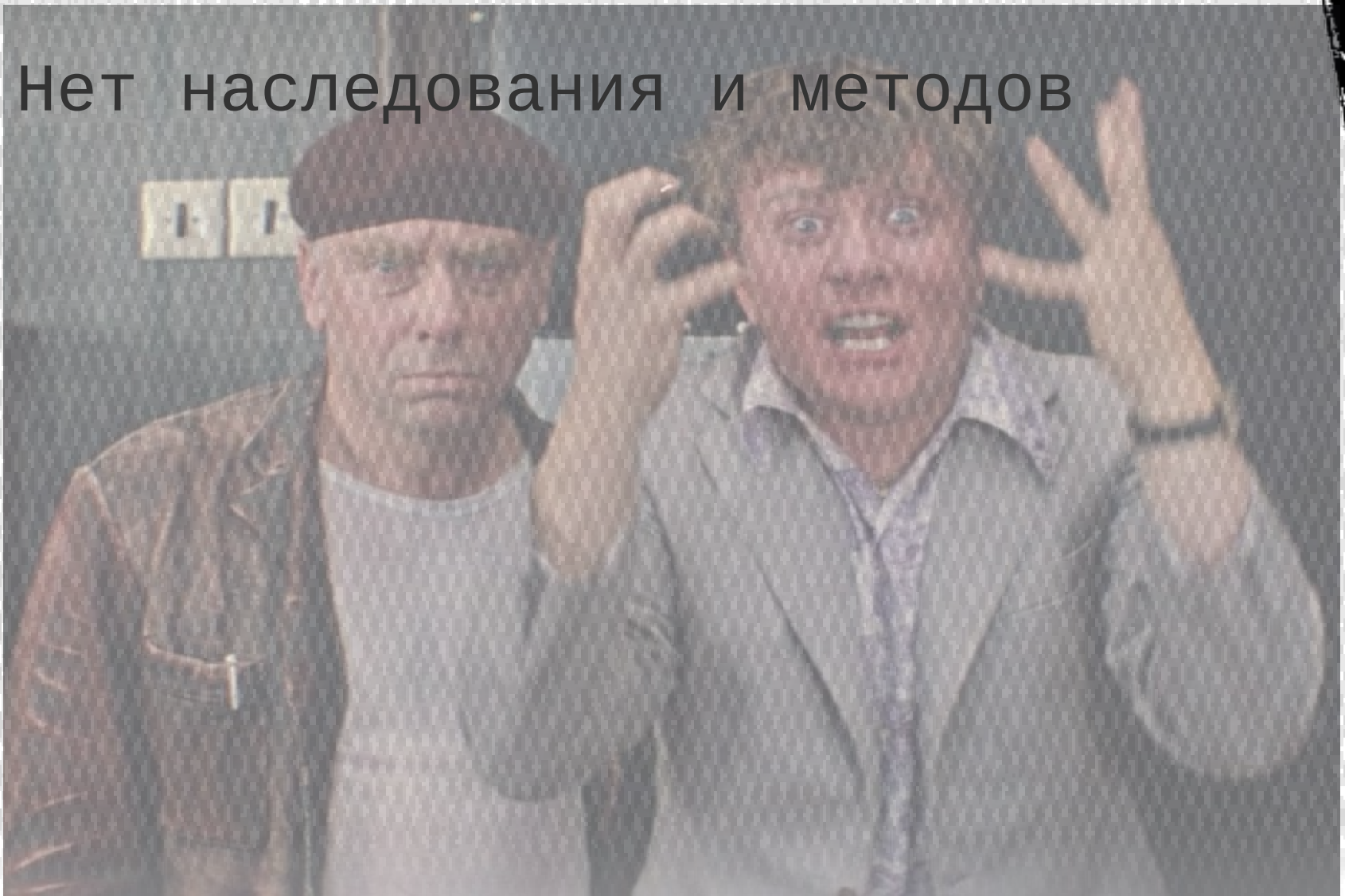
Высокоуровневый язык



- Язык Си стар как твоя мамка
  - да, 1969г.
- Изначально создавался для Unix
  - Первая версия Unix была написана на языке ассемблера
- В нём не было структур, unsigned и void
  - зато был оператор **=+**

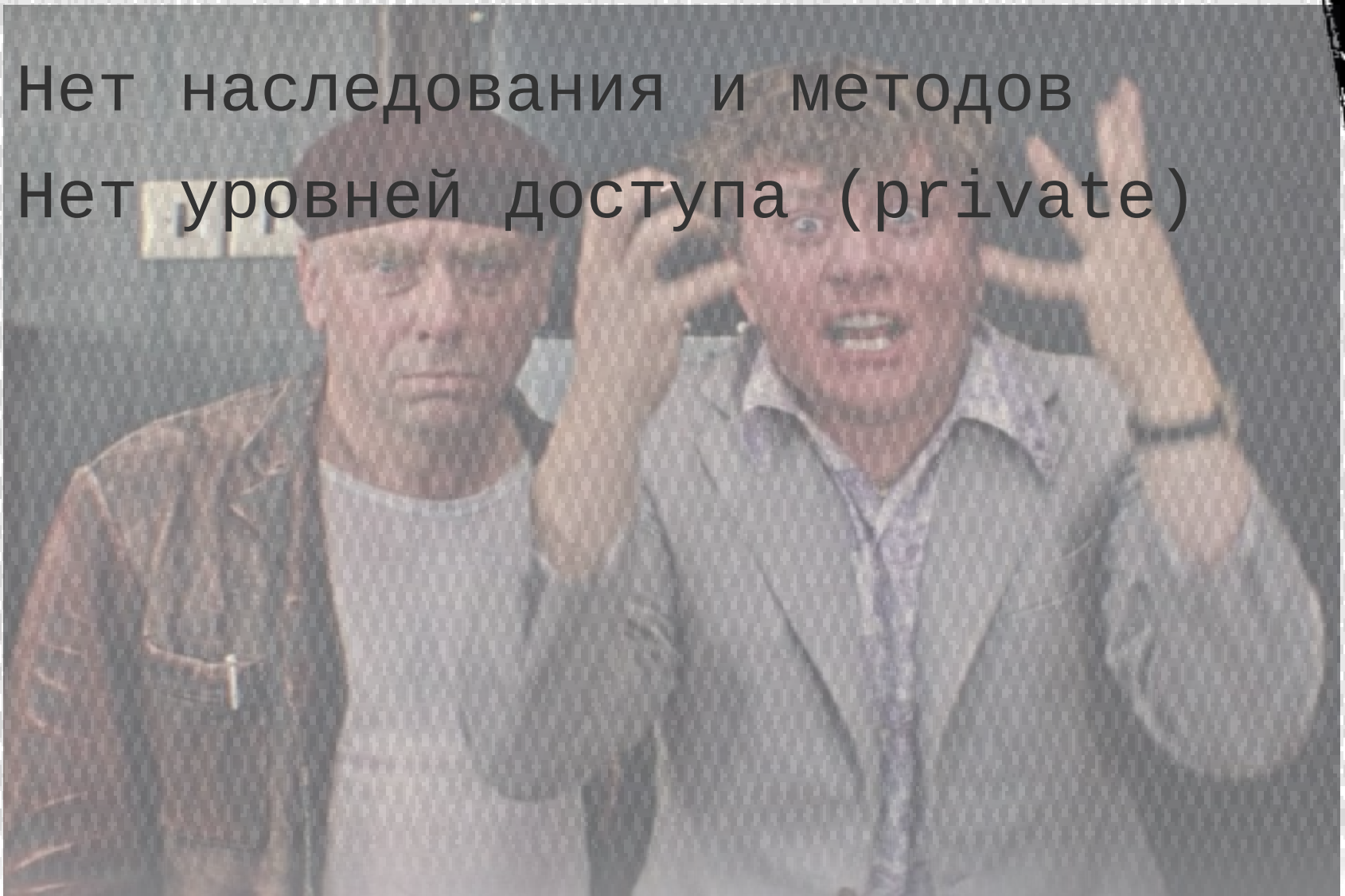
Высокоуровневый язык

- Нет наследования и методов

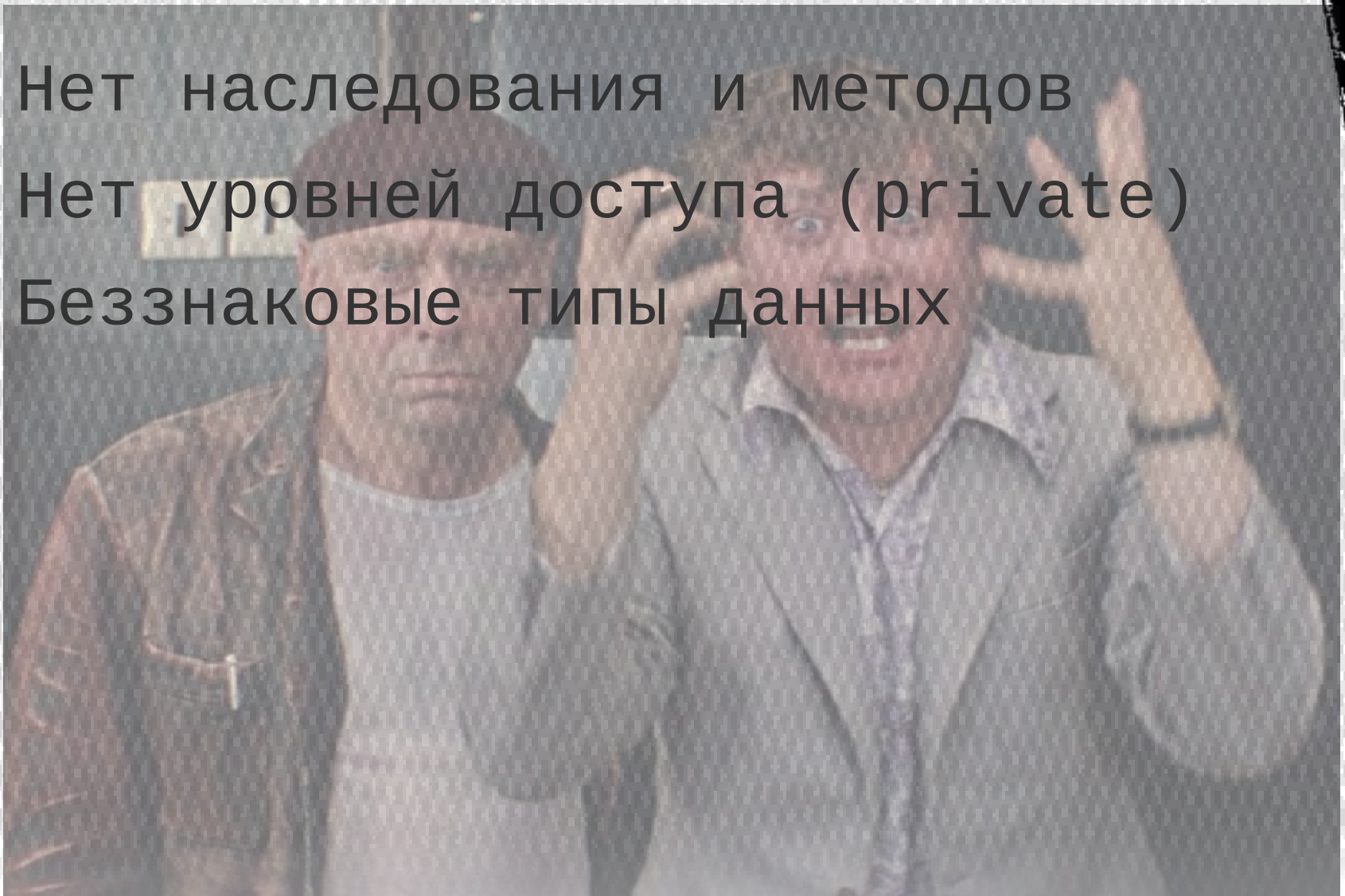


Высокоуровневый язык

- Нет наследования и методов
- Нет уровней доступа (private)

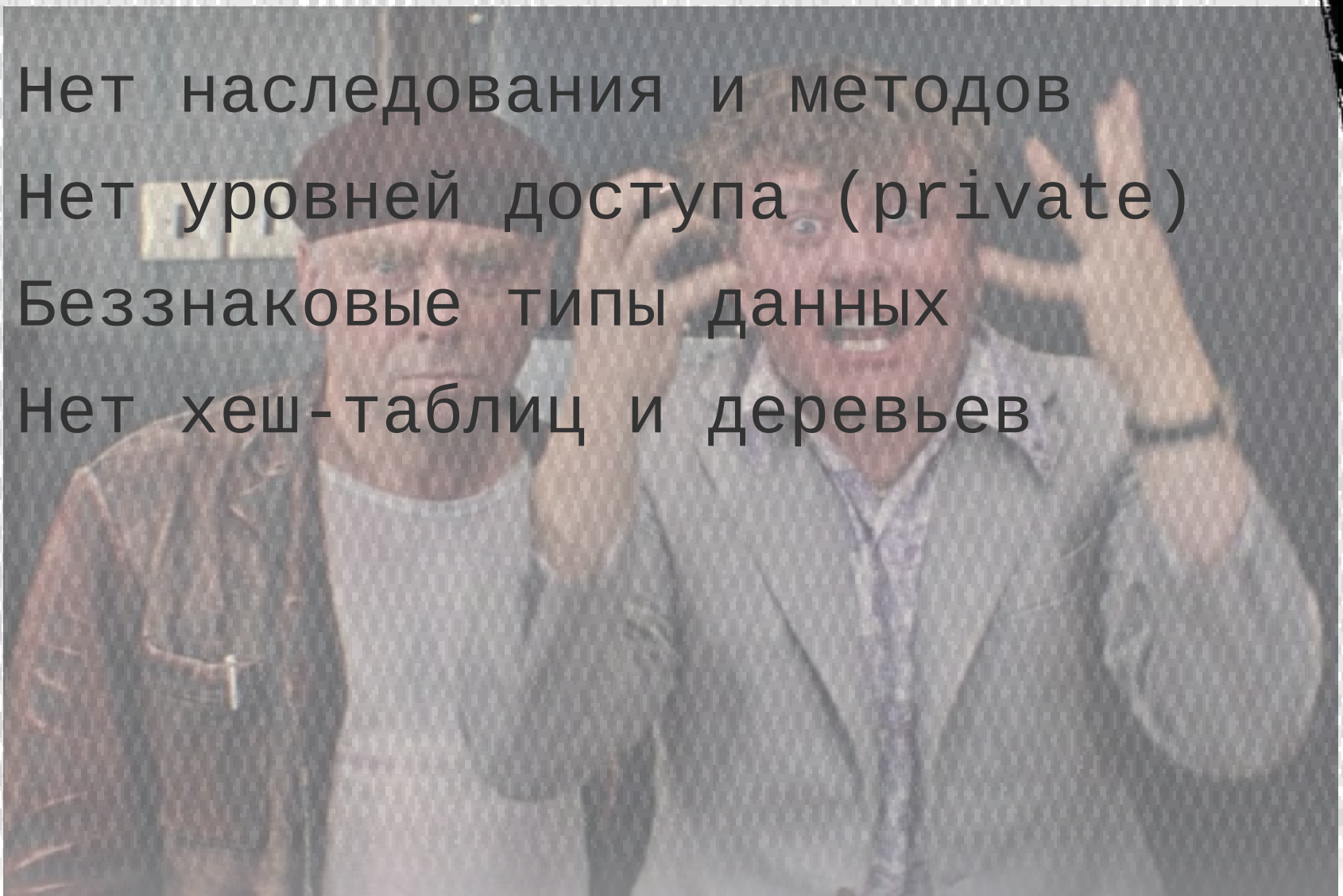


Высокоуровневый язык



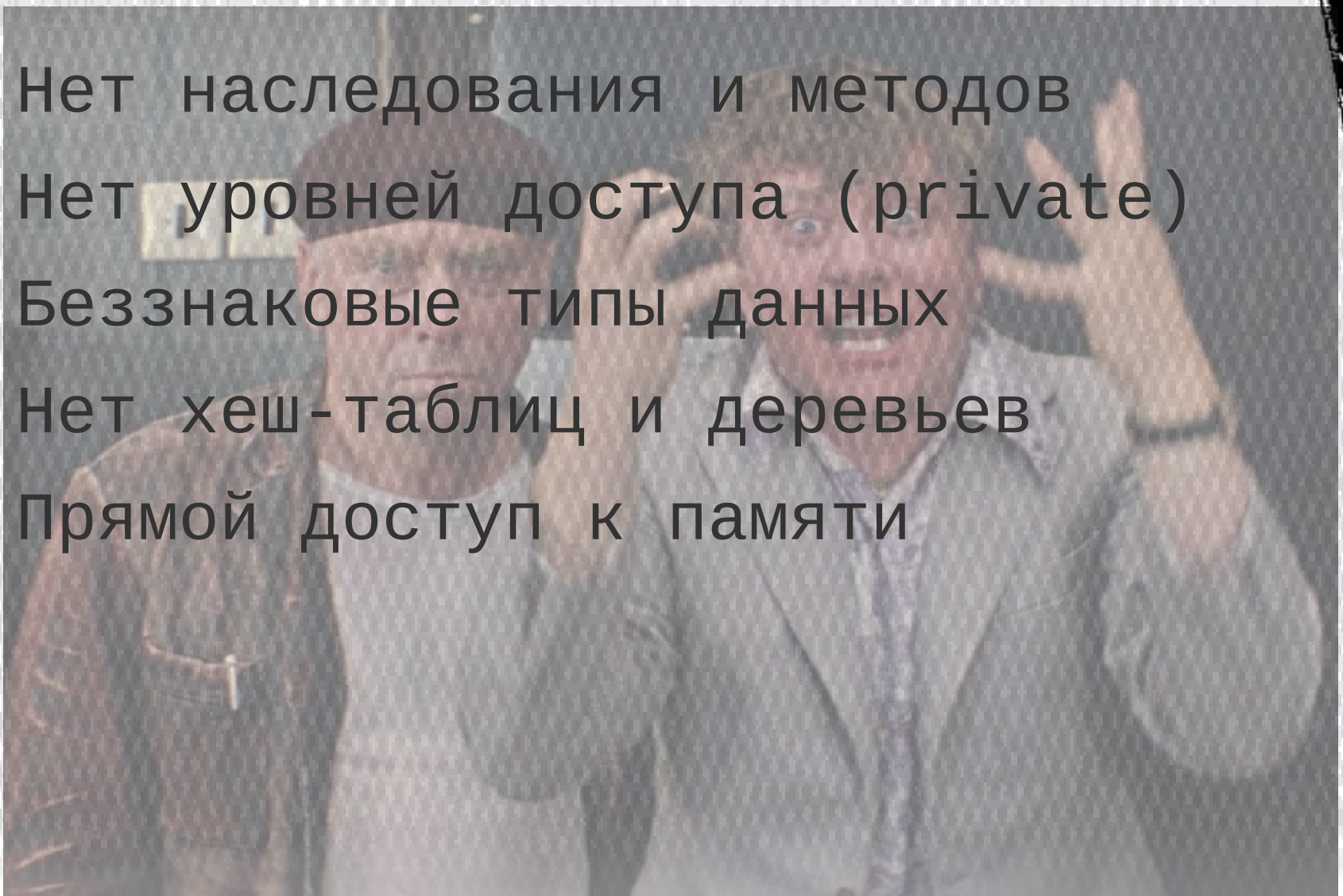
- Нет наследования и методов
- Нет уровней доступа (private)
- Беззнаковые типы данных

Высокоуровневый язык



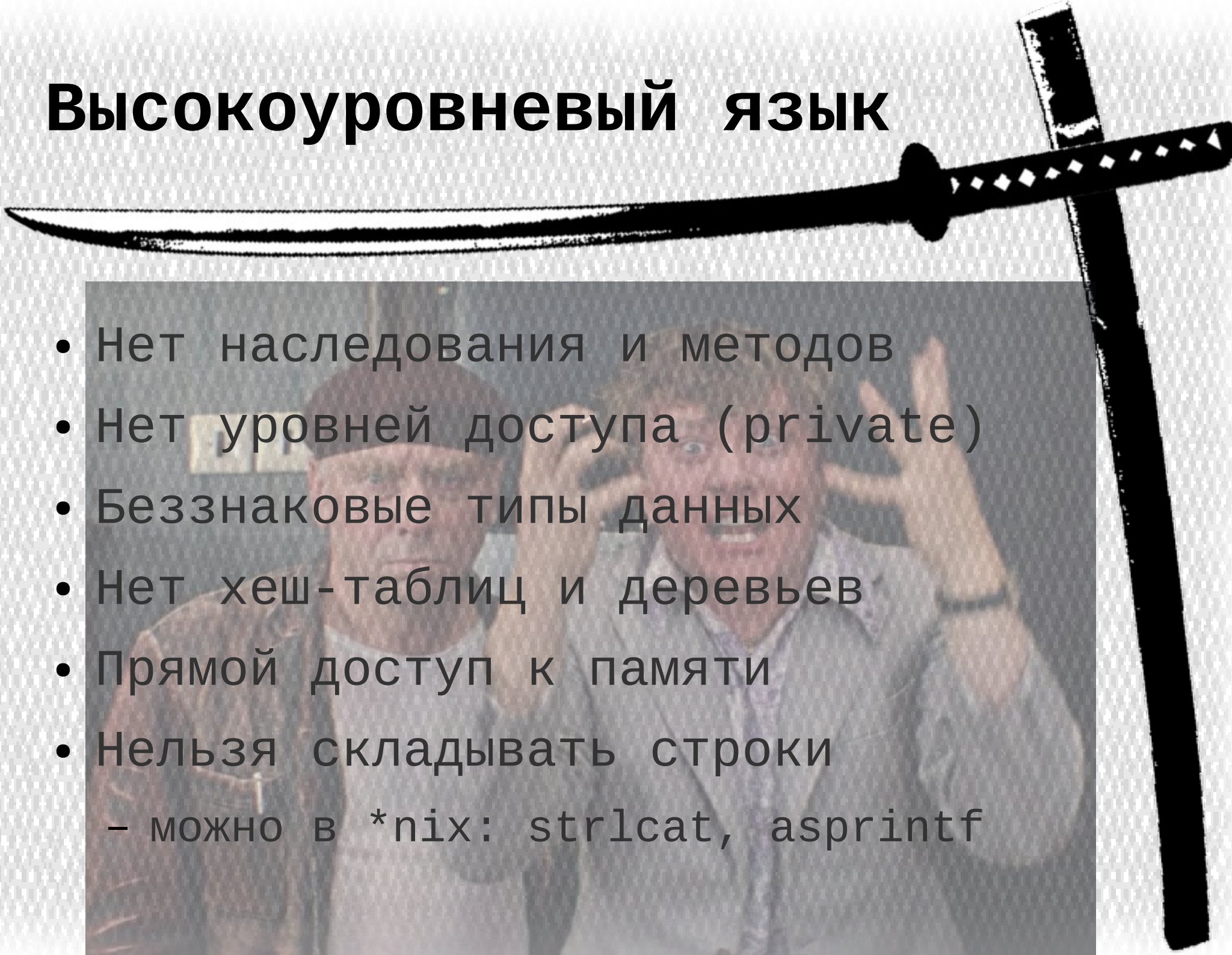
- Нет наследования и методов
- Нет уровней доступа (private)
- Беззнаковые типы данных
- Нет хеш-таблиц и деревьев

Высокоуровневый язык



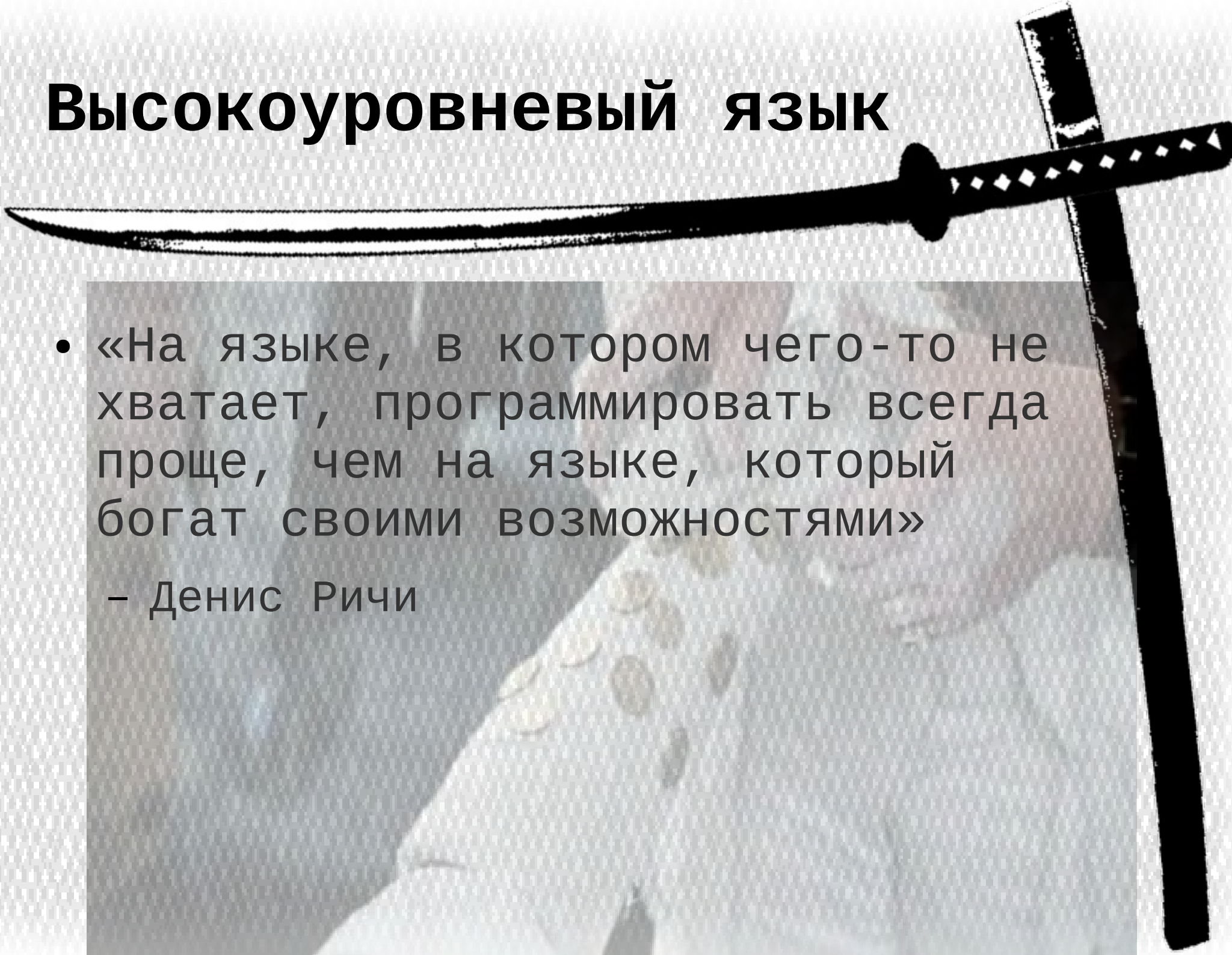
- Нет наследования и методов
- Нет уровней доступа (private)
- Беззнаковые типы данных
- Нет хеш-таблиц и деревьев
- Прямой доступ к памяти

Высокоуровневый язык



- Нет наследования и методов
- Нет уровней доступа (private)
- Беззнаковые типы данных
- Нет хеш-таблиц и деревьев
- Прямой доступ к памяти
- Нельзя складывать строки
  - можно в *nix: strlcat, asprintf

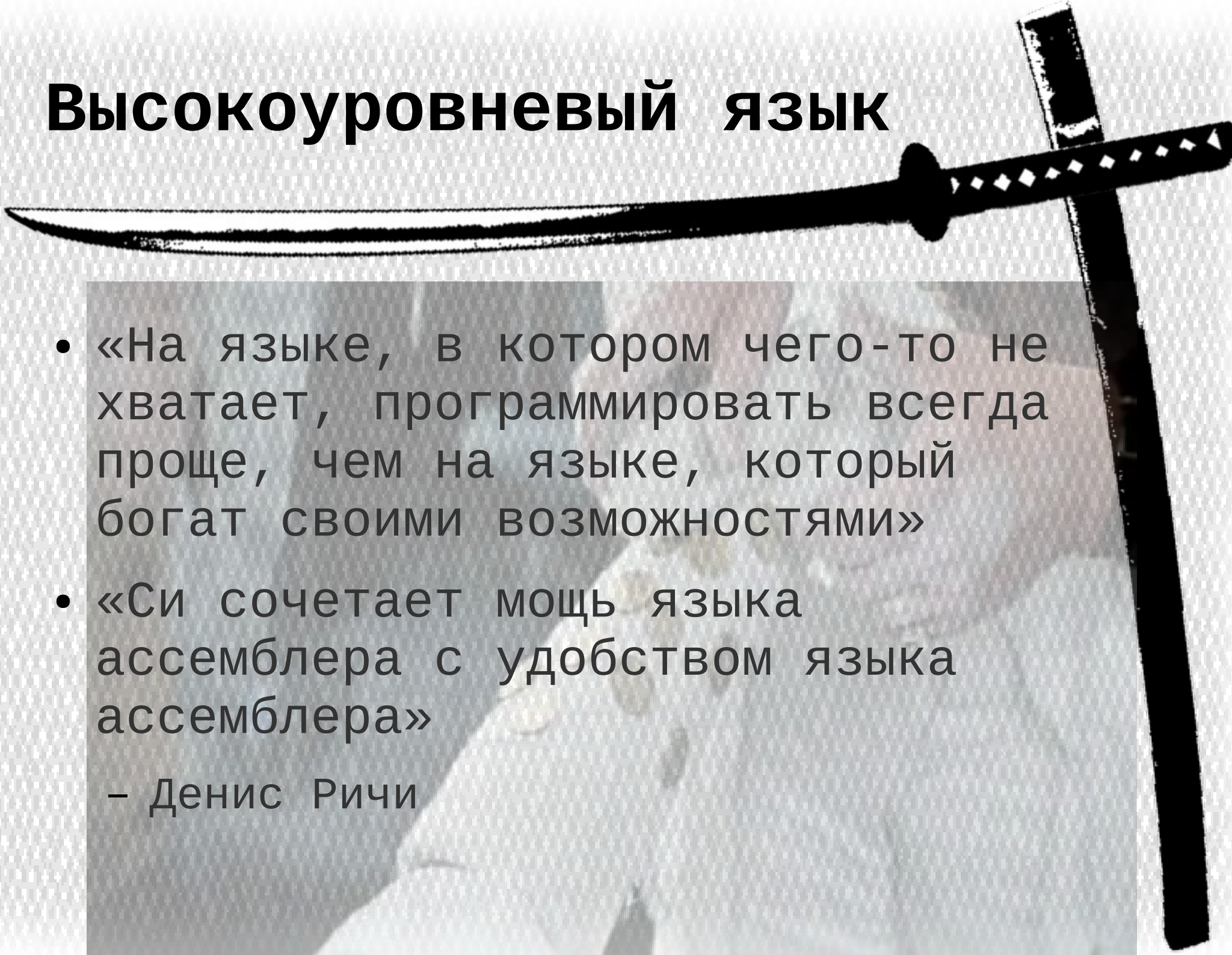
Высокоуровневый язык



- «На языке, в котором чего-то не хватает, программировать всегда проще, чем на языке, который богат своими возможностями»

  – Денис Ричи

Высокоуровневый язык



- «На языке, в котором чего-то не хватает, программировать всегда проще, чем на языке, который богат своими возможностями»
- «Си сочетает мощь языка ассемблера с удобством языка ассемблера»

  – Денис Ричи

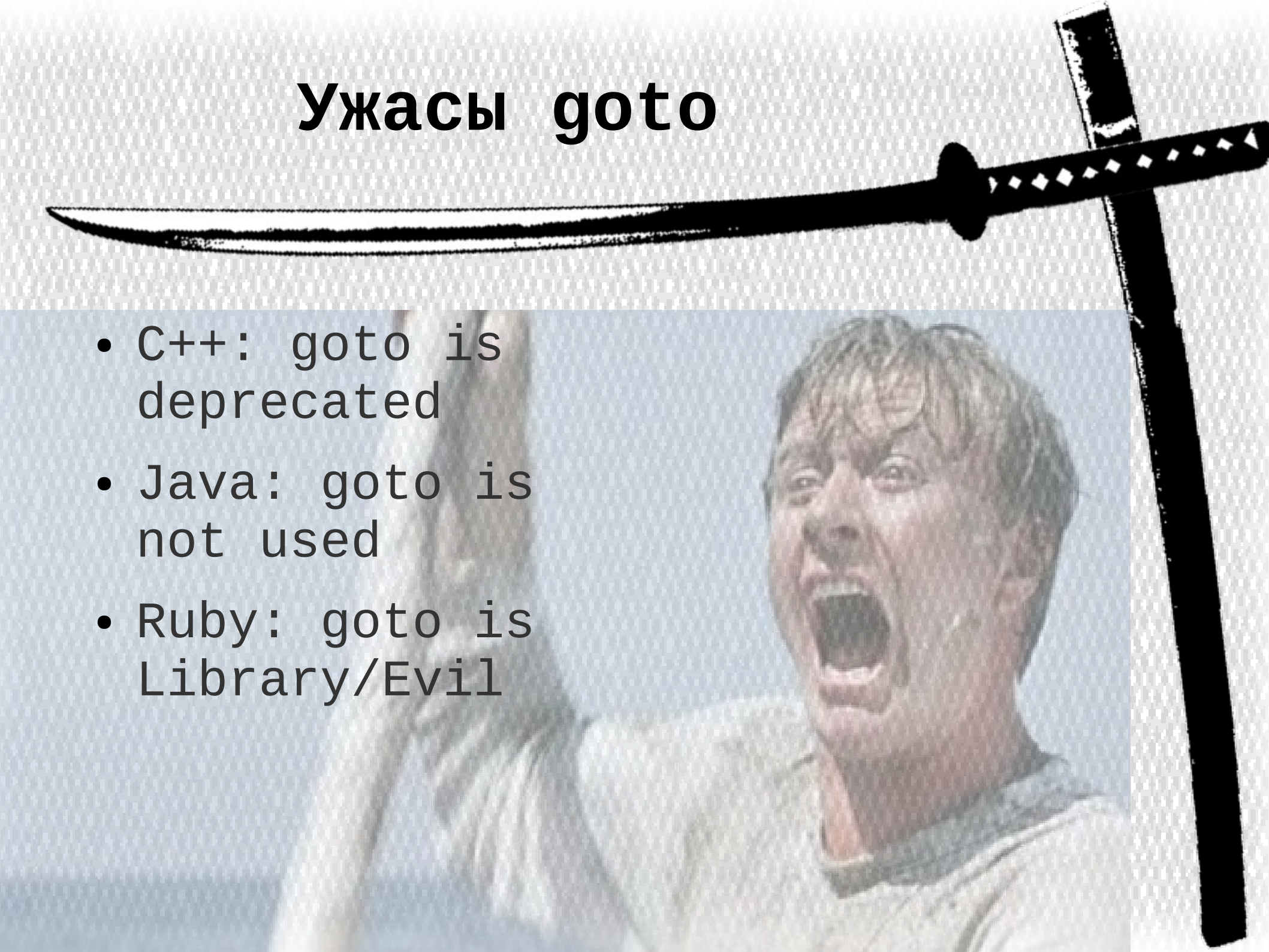
Ужасы goto

- C++: goto is deprecated

Ужасы goto

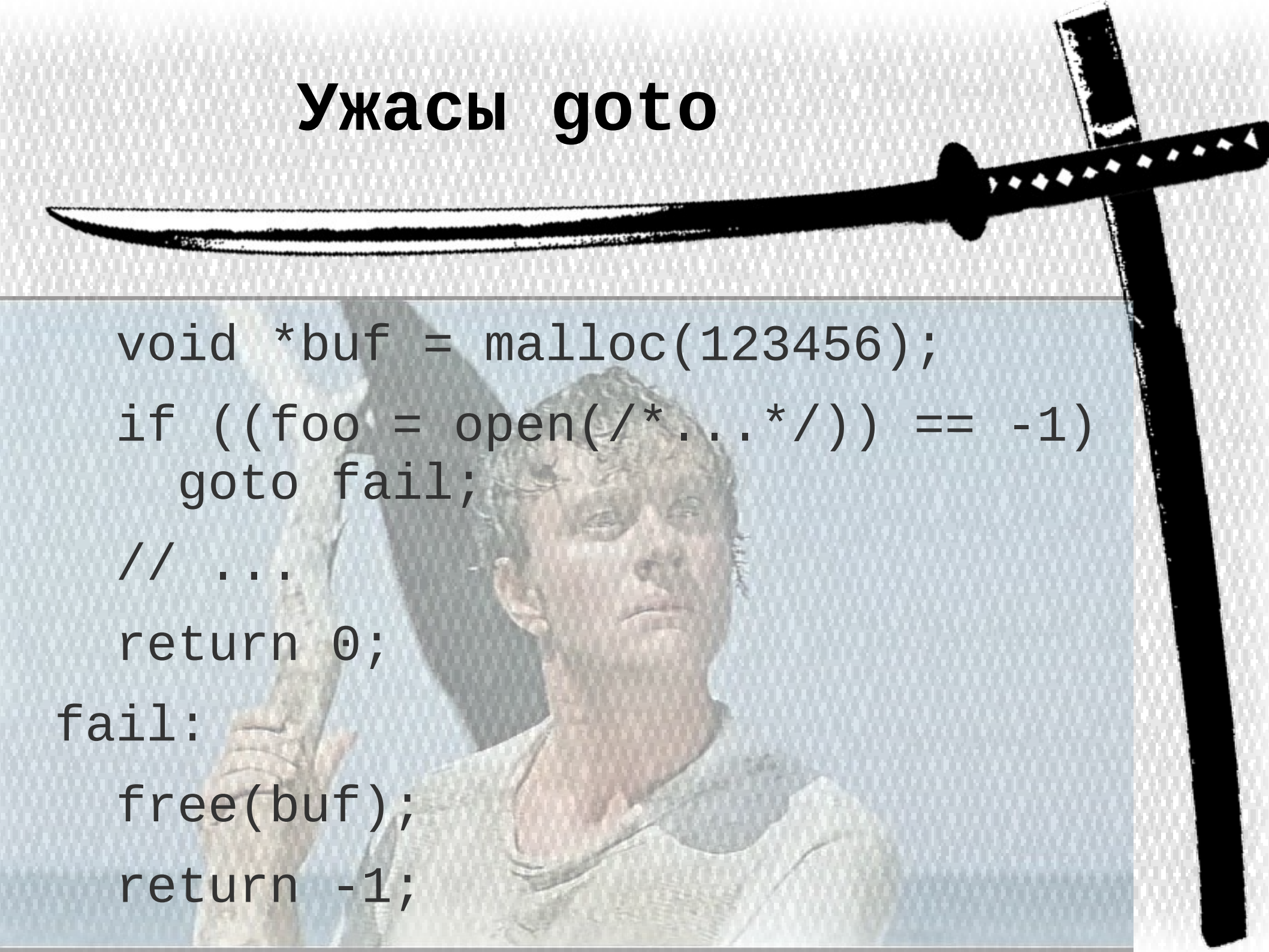
- C++: goto is deprecated
- Java: goto is not used

Ужасы goto



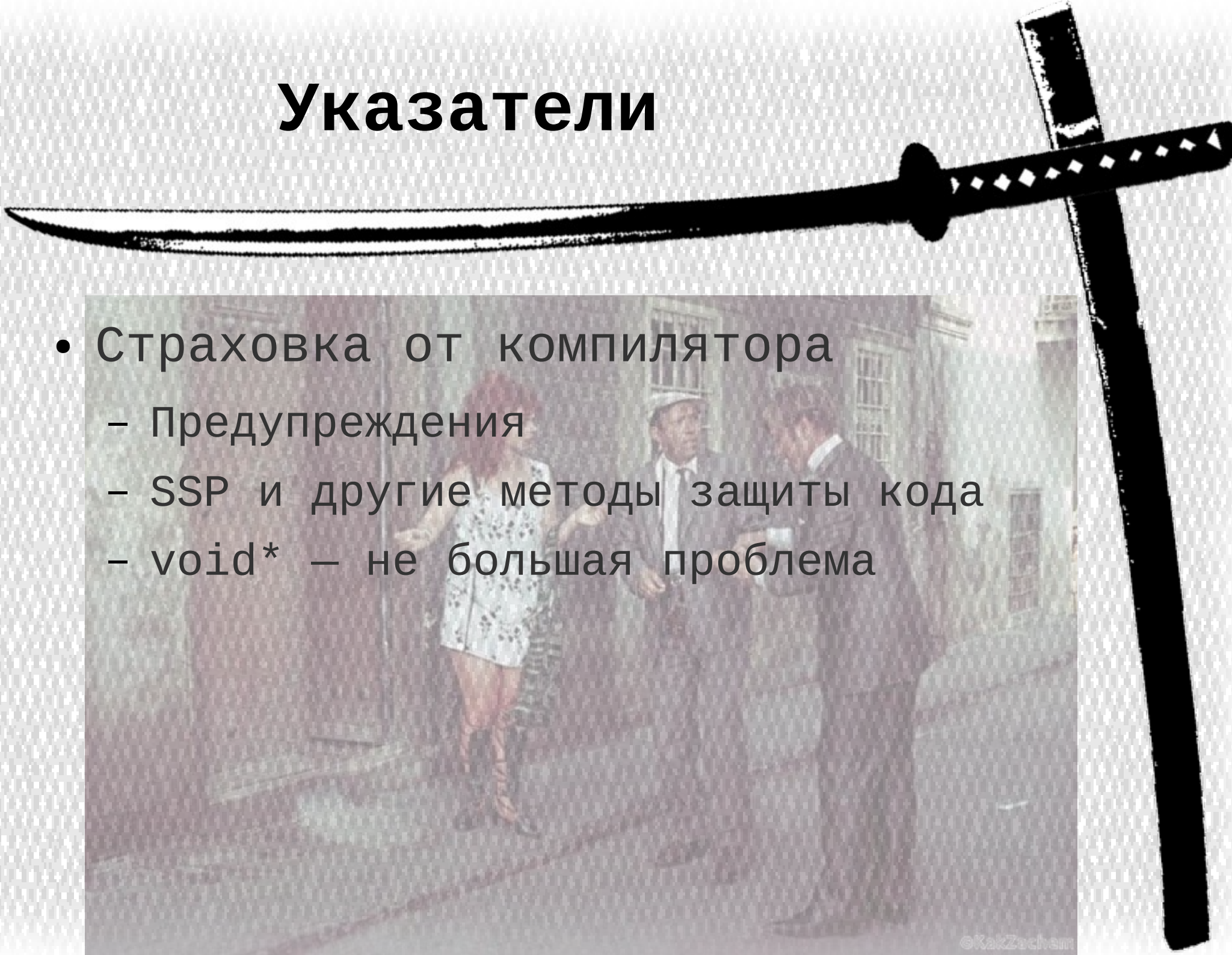
- C++: goto is deprecated
- Java: goto is not used
- Ruby: goto is Library/Evil

Ужасы goto



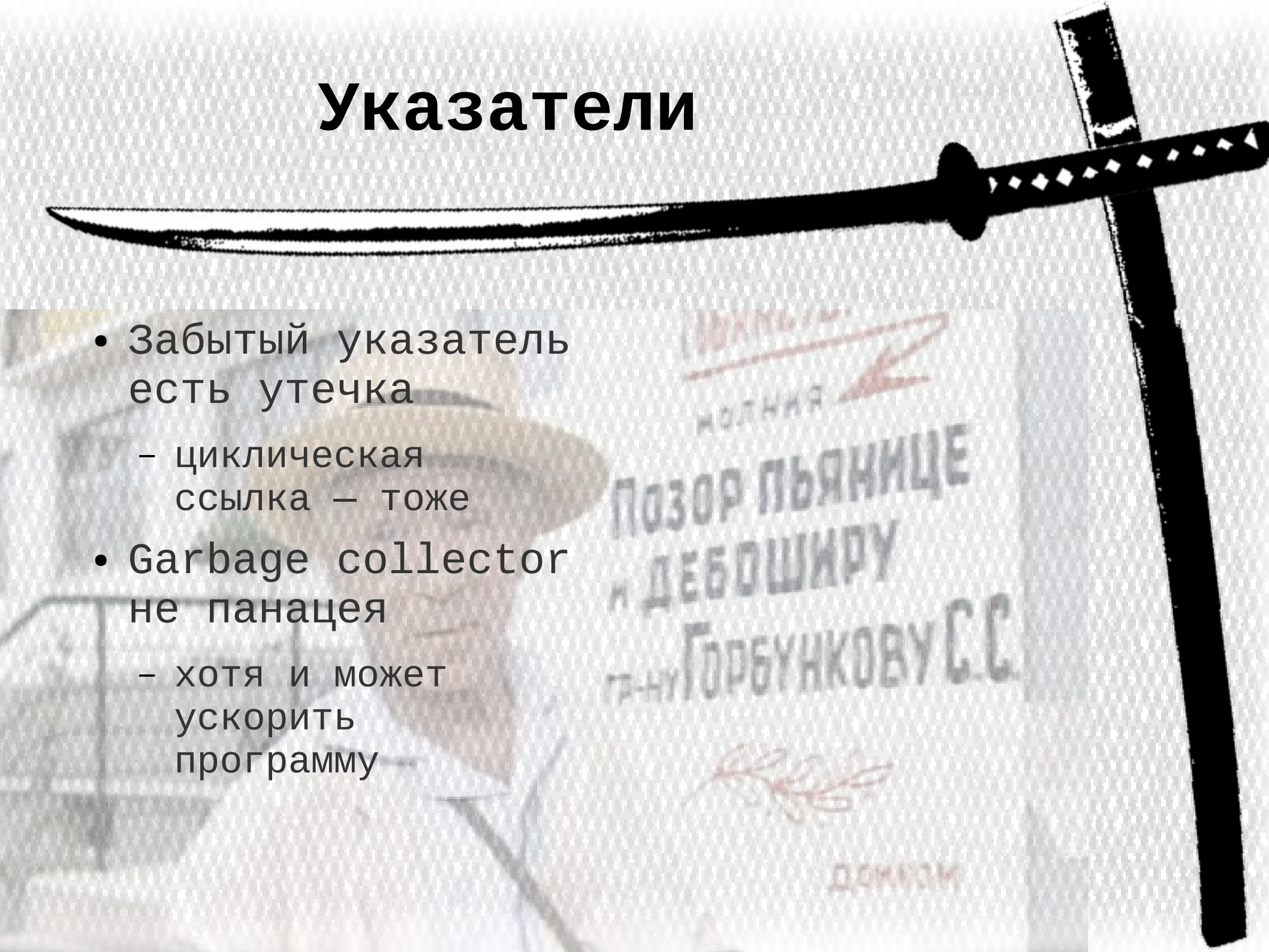
```
  void *buf = malloc(123456);
  if ((foo = open(/*...*/)) == -1)
    goto fail;
  // ...
  return 0;
fail:
  free(buf);
  return -1;
```

Указатели



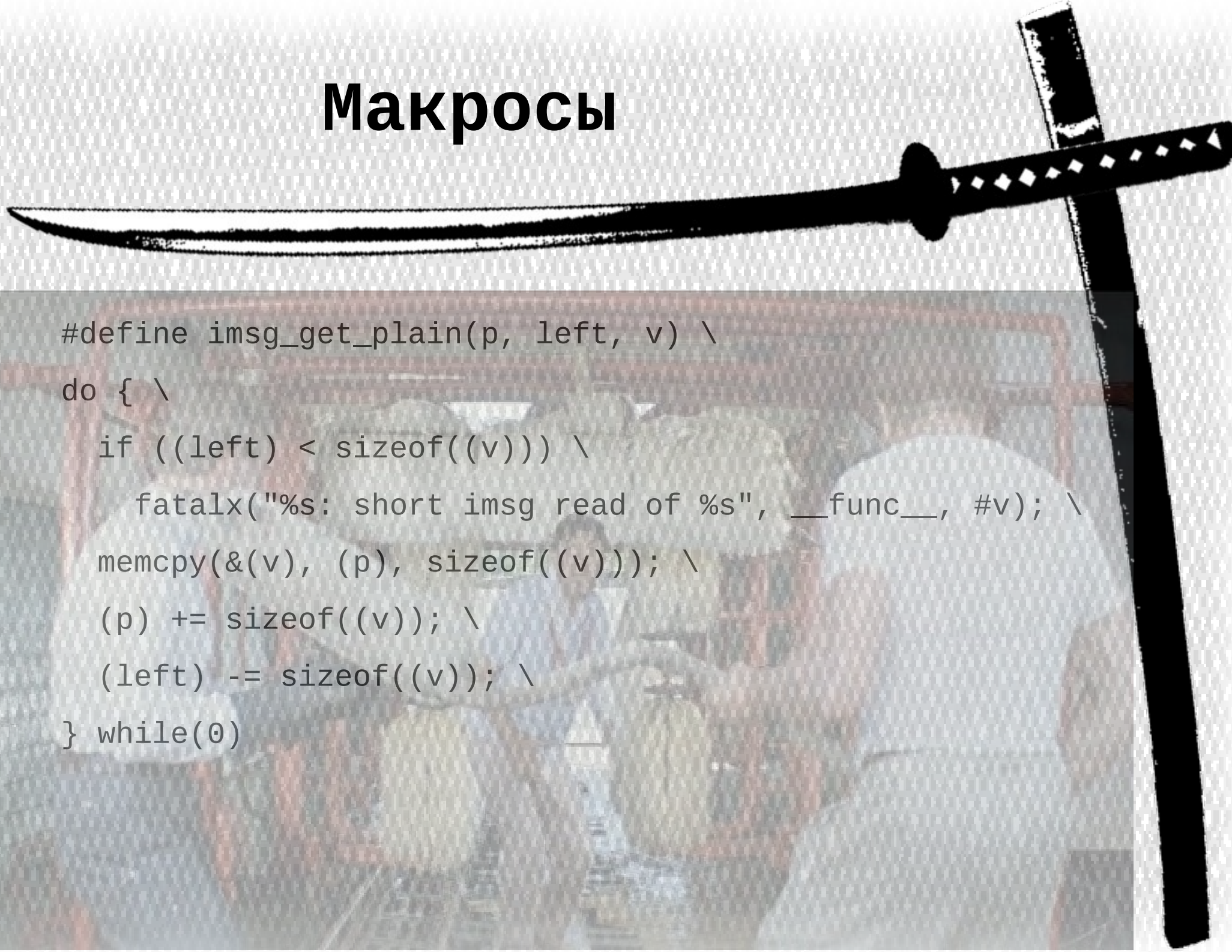
- Страховка от компилятора
  - Предупреждения
  - SSP и другие методы защиты кода
  - void* — не большая проблема

Указатели



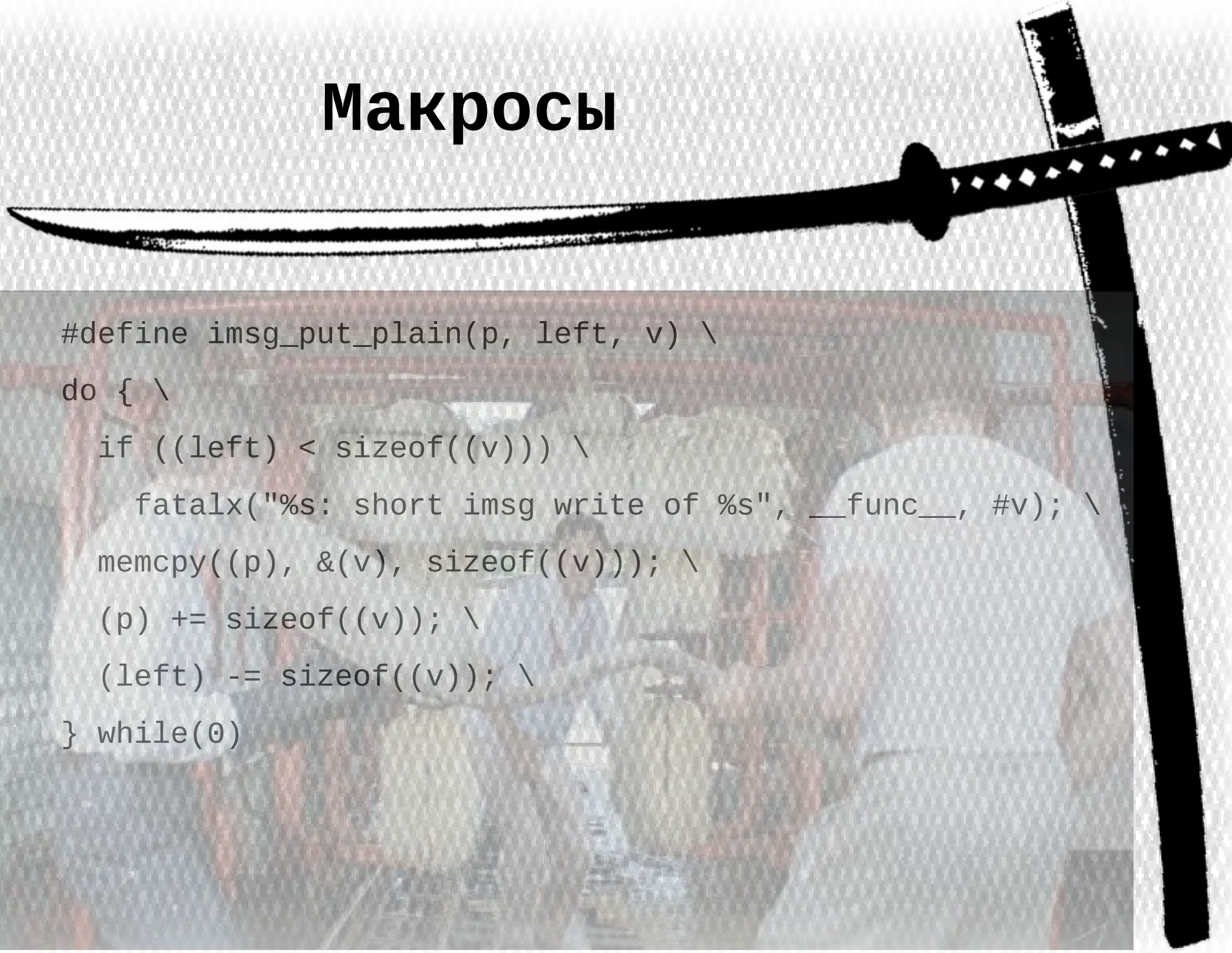
- Забытый указатель есть утечка
  - циклическая ссылка — тоже
- Garbage collector не панацея
  - хотя и может ускорить программу

Макросы



```
#define imsg_get_plain(p, left, v) \
do { \
  if ((left) < sizeof((v))) \
    fatalx("%s: short imsg read of %s", __func__, #v); \
  memcpy(&(v), (p), sizeof((v))); \
  (p) += sizeof((v)); \
  (left) -= sizeof((v)); \
} while(0)
```

Макросы

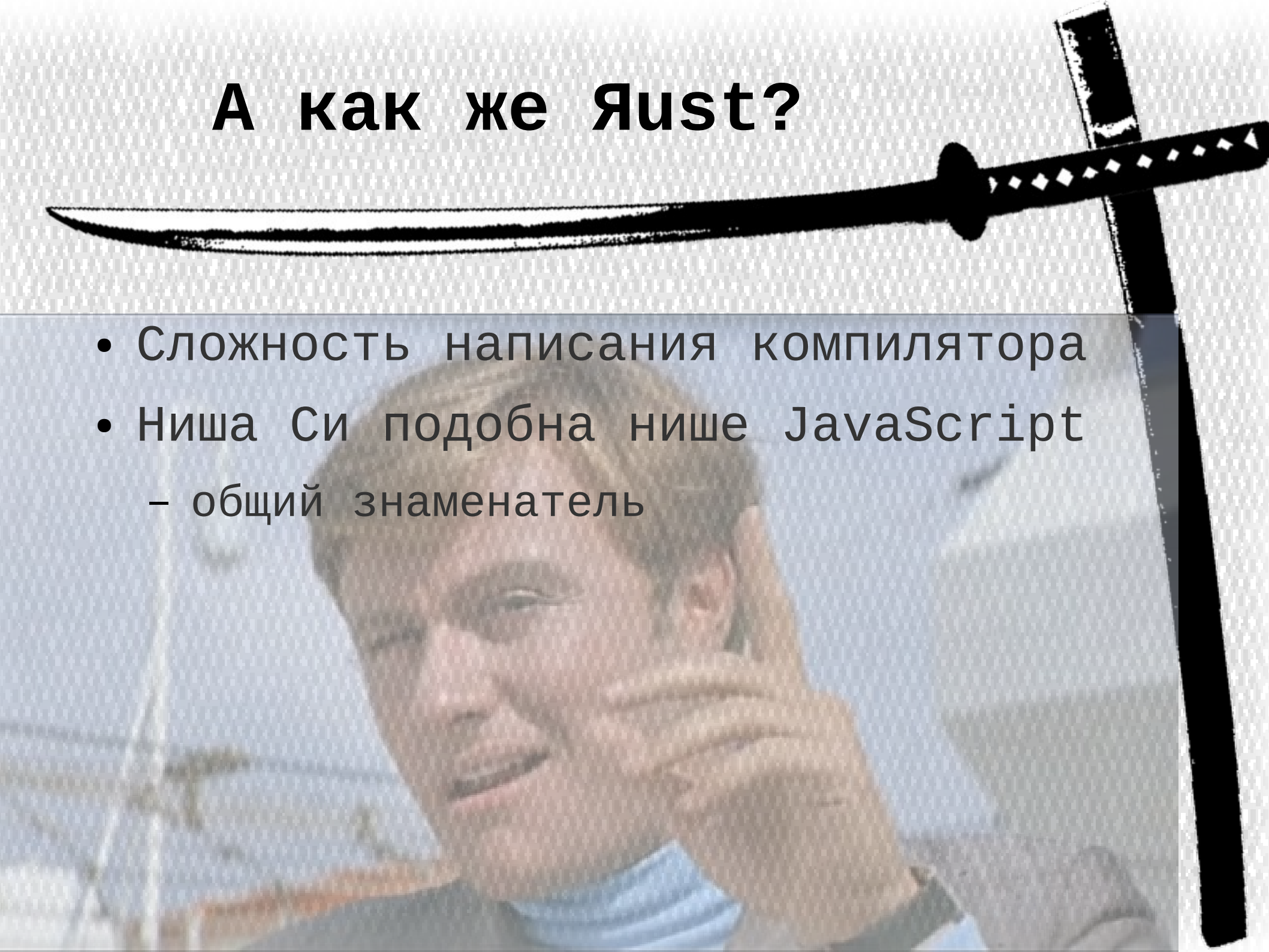


```
#define imsg_put_plain(p, left, v) \
do { \
  if ((left) < sizeof((v))) \
    fatalx("%s: short imsg write of %s", __func__, #v); \
  memcpy((p), &(v), sizeof((v))); \
  (p) += sizeof((v)); \
  (left) -= sizeof((v)); \
} while(0)
```

Макросы

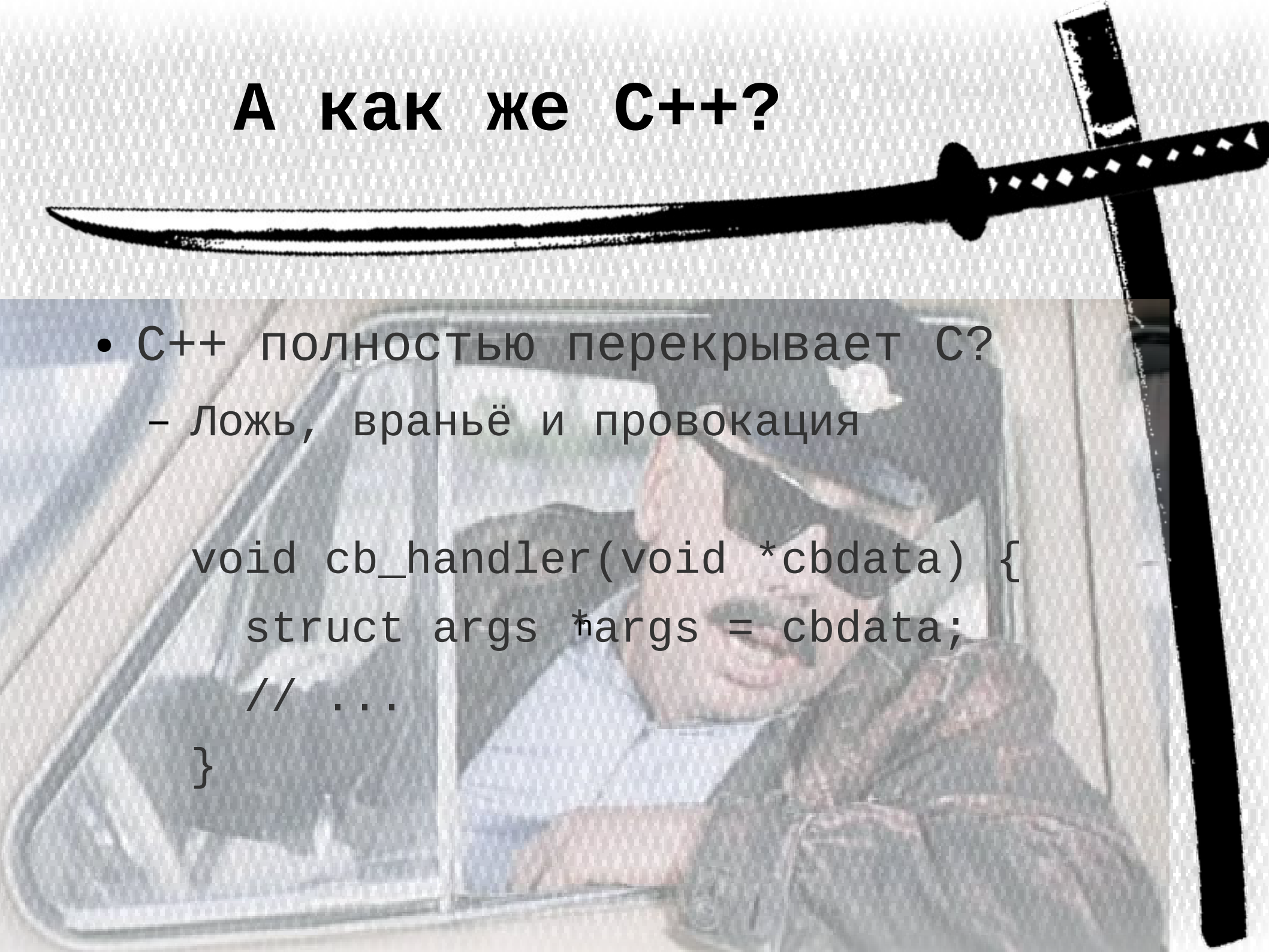
```
imsg_put_plain(buf, buflen, frag->fr_id);
imsg_put_plain(buf, buflen, frag->fr_from);
imsg_put_plain(buf, buflen, frag->fr_till);
// ...
imsg_get_plain(buf, buflen, frag->fr_id);
imsg_get_plain(buf, buflen, frag->fr_from);
imsg_get_plain(buf, buflen, frag->fr_till);
```

А как же Rust?



- Сложность написания компилятора
- Ниша Си подобна нише JavaScript
  - общий знаменатель

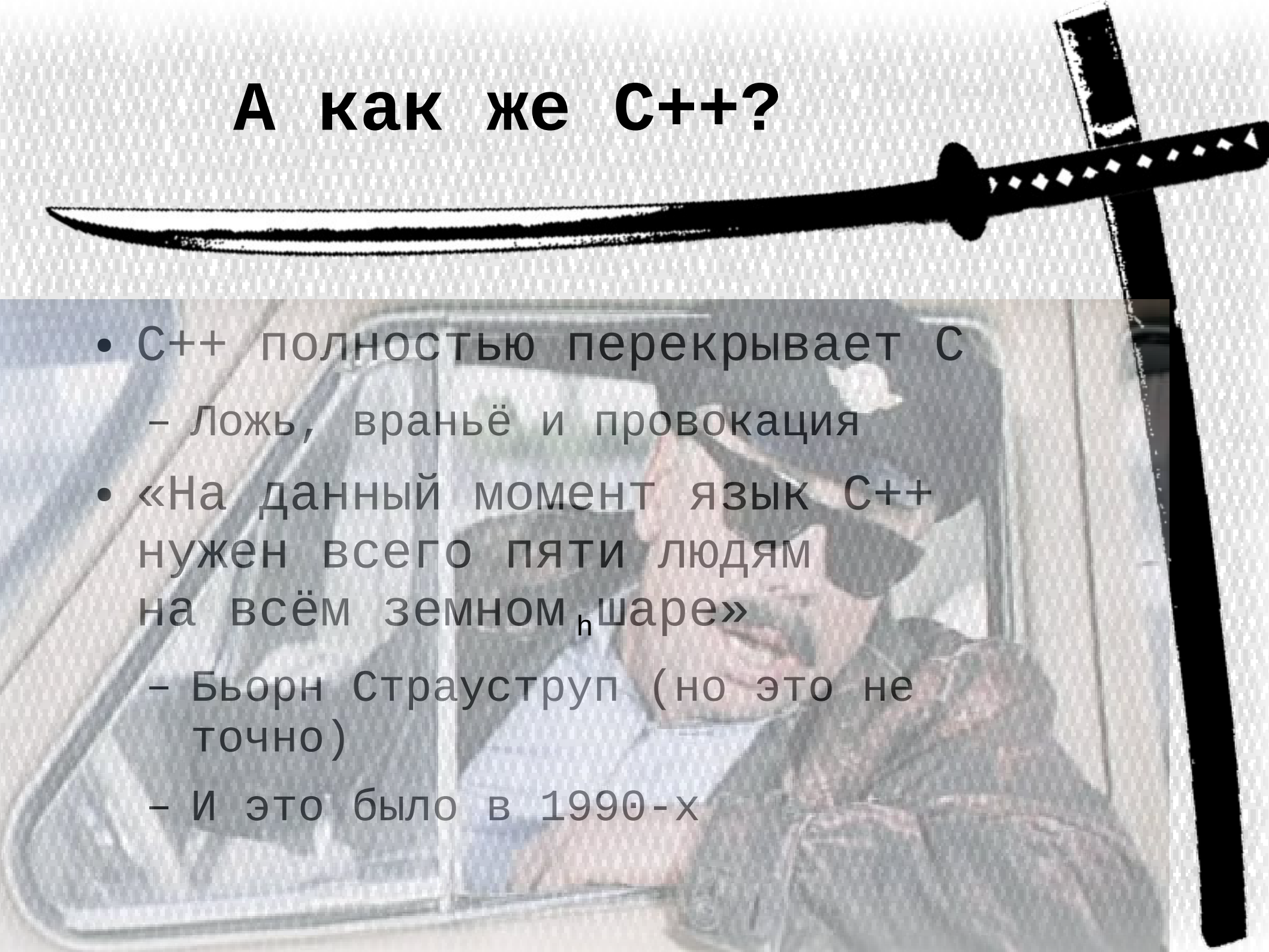
А как же C++?



- C++ полностью перекрывает C?
  - Ложь, враньё и провокация

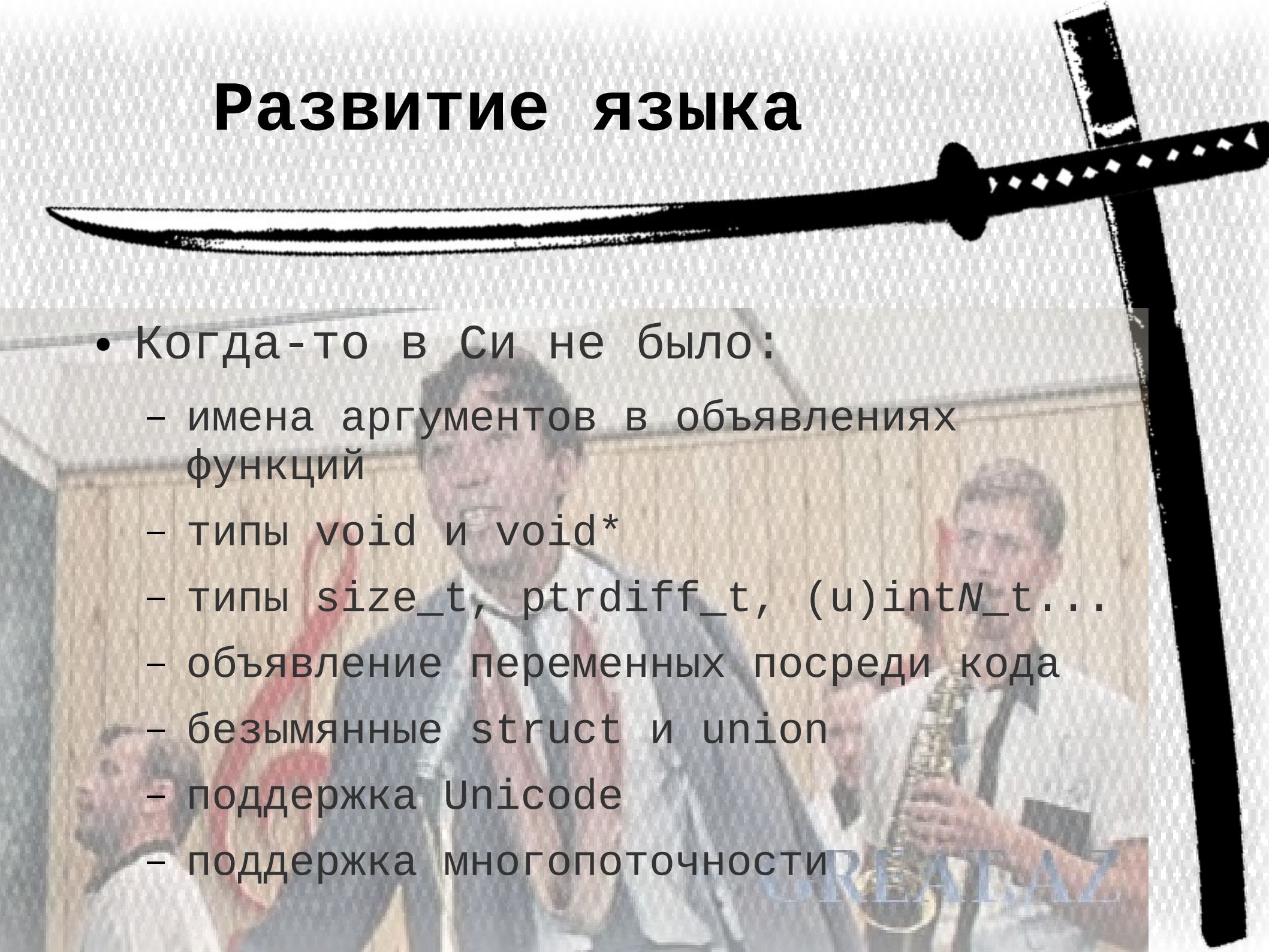
```
void cb_handler(void *cbdata) {
  struct args *args = cbdata;
  // ...
}
```

А как же C++?



- C++ полностью перекрывает C
  - Ложь, враньё и провокация
- «На данный момент язык C++ нужен всего пяти людям на всём земном$_h$шаре»
  - Бьорн Страуструп (но это не точно)
  - И это было в 1990-х

Развитие языка

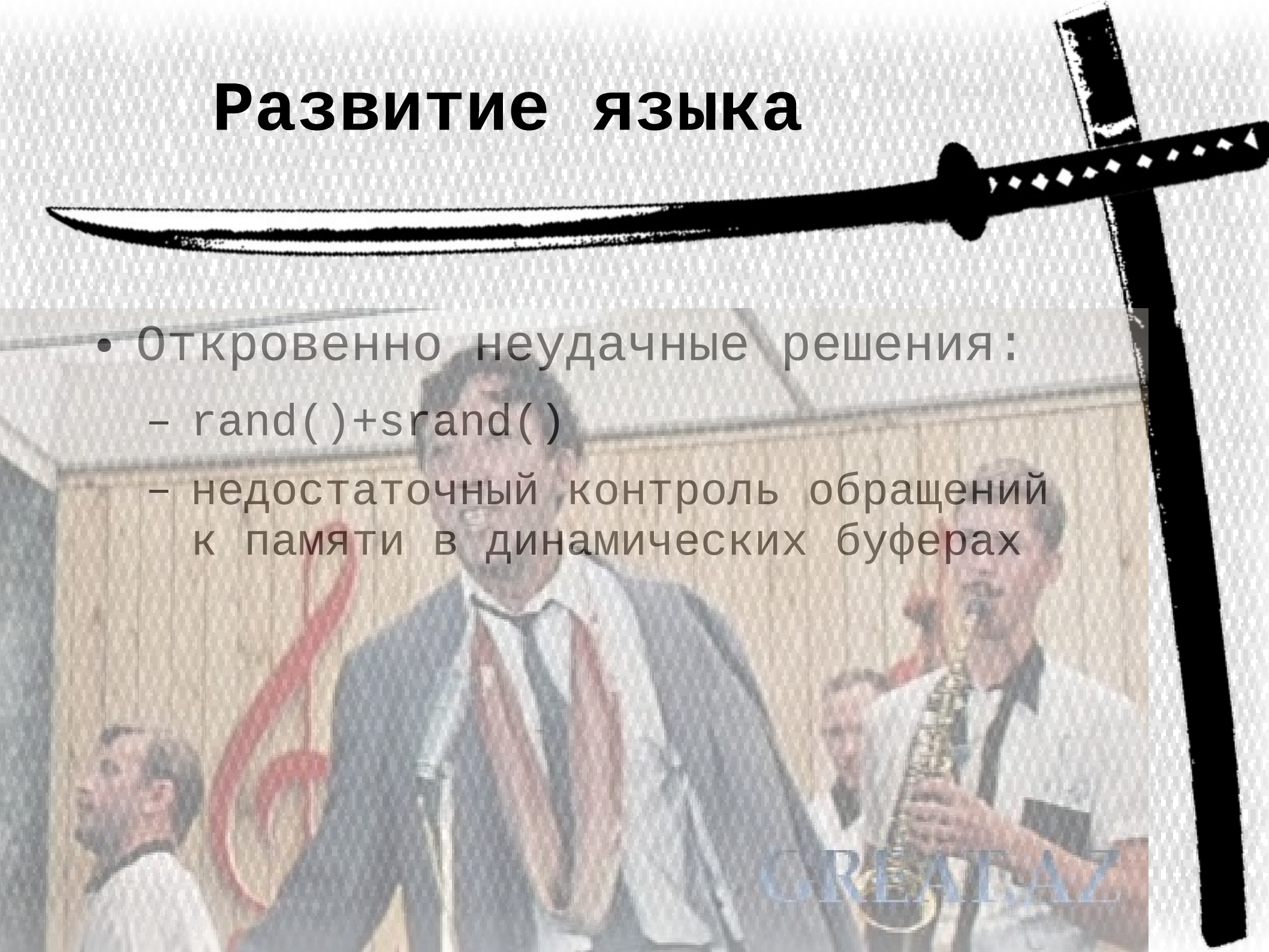


- Когда-то в Си не было:
  - имена аргументов в объявлениях функций
  - типы void и void*
  - типы size_t, ptrdiff_t, (u)int*N*_t...
  - объявление переменных посреди кода
  - безымянные struct и union
  - поддержка Unicode
  - поддержка многопоточности

Развитие языка

- Не очень удачные решения:
  - strict aliasing
  - Exit(), quickexit()

Развитие языка



- Откровенно неудачные решения:
  - rand()+srand()
  - недостаточный контроль обращений к памяти в динамических буферах

Вопросы?

Канец

Спасибо!