TestCon Moscow 2019

Статический анализ как дополнительный барьер на пути ошибок

Сергей Хренов

PVS-Studio

“У нас были включены все уровни предупреждений компилятора, 98% кода покрыто тестами, три тим-лида проводили Code-Review, мы даже использовали парную разработку и Agile. Не то, чтобы всё это было нужно в работе, но раз взялся делать код надёжным, то иди в этом до конца.”

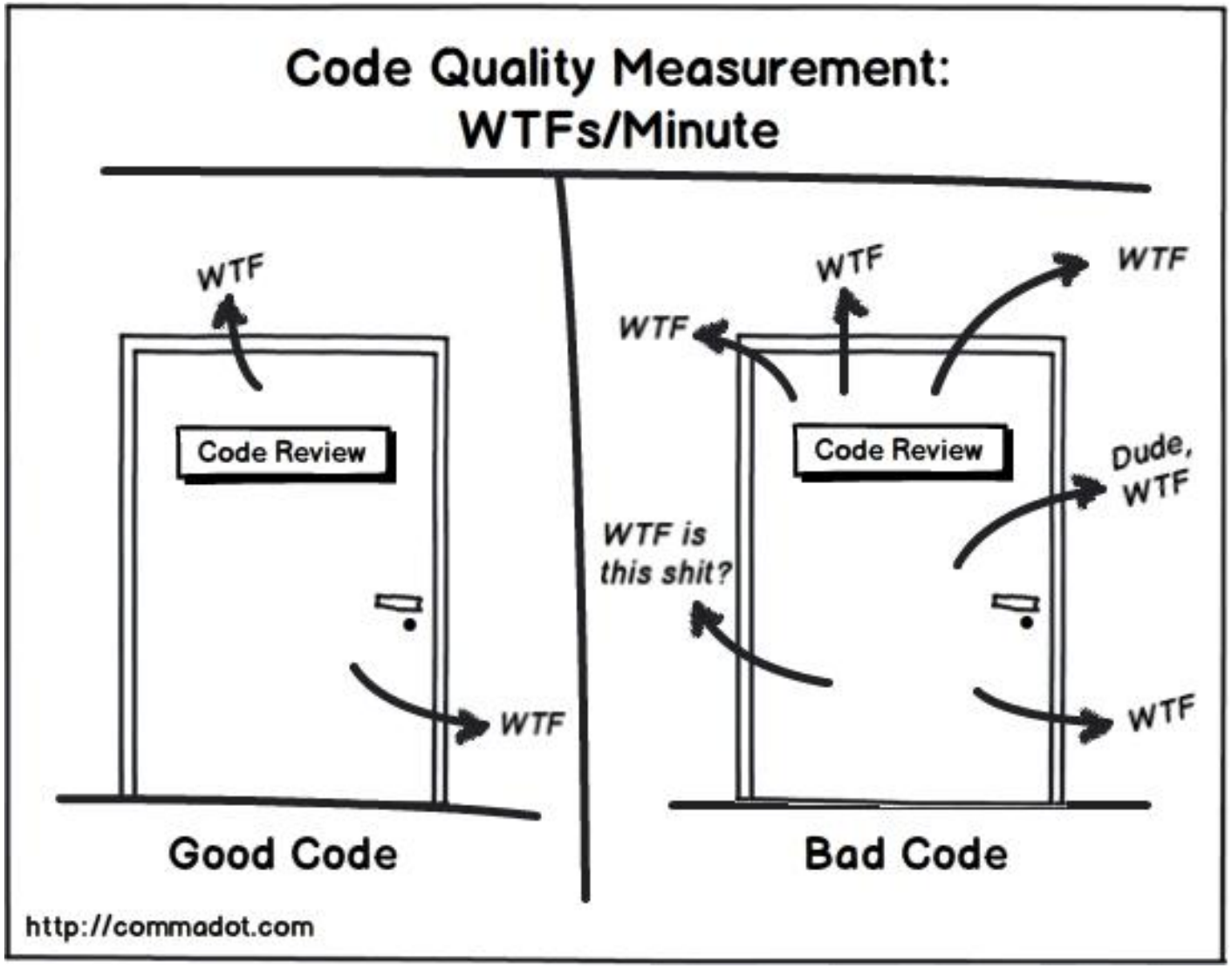
Проблематика

- Объём кодовой базы растёт
- При этом плотность ошибок растёт нелинейно
- Требования к коду всё выше
- Старых методов контроля качества уже недостаточно

Методы улучшения качества кода

- Делать сразу правильно (не работает)
- Следование корпоративным правилам
- Использование “лучших практик”
- Парная разработка
- Разработка через тестирование (TDD)
- Гибкая разработка Agile
- Совместные обзоры кода
- Инструментальные средства

Обзоры кода


Code Quality Measurement:
WTFs/Minute
WTF
Code Review
WTF
Good Code
WTF
WTF
WTF
WTF
Code Review
Dude,
WTF
WTF is
this shit?
WTF
Bad Code
http://commadot.com

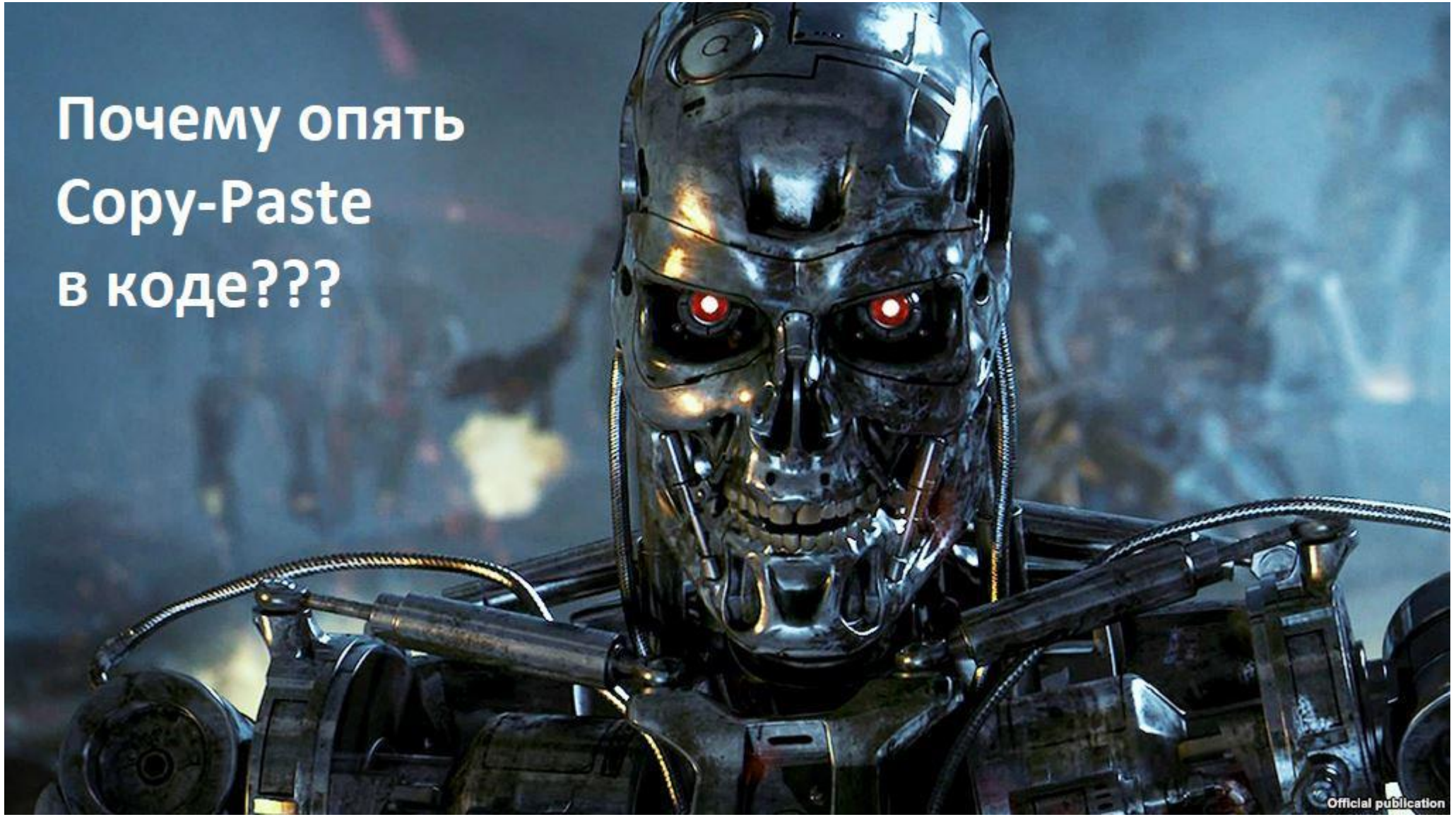
Используй инструментальные средства

- Нагрузочные тесты
- Unit-тесты
- Динамические анализаторы
- **Статические анализаторы**
- ???
- PROFIT!!!

Статический анализ кода

- Не заменяет, но дополняет обзоры кода
- Позволяет контролировать качество кода в больших проектах
- Рассмотрим, что такое статический анализ кода
- Для чего ещё он может использоваться

Статический анализ – это…



Статический анализ кода

Преимущества:

- Раннее обнаружение проблем
- Максимальное покрытие кода
- Поиск разнообразных паттернов ошибок

Недостатки:

- Ложные срабатывания
- Неизвестна точная критичность ошибки

Зачем нужен статический анализ кода

```
button_checked_gradient_begin = use_system_colors ? Color.Empty : Color.FromArgb (255, 223, 154);
button_checked_gradient_end = use_system_colors ? Color.Empty : Color.FromArgb (255, 166, 76);
button_checked_gradient_middle = use_system_colors ? Color.Empty : Color.FromArgb (255, 195, 116);
button_checked_highlight = Color.FromArgb (195, 211, 237);
button_checked_highlight_border = Color.FromKnownColor (KnownColor.Highlight);
button_pressed_border = use_system_colors ? Color.FromKnownColor (KnownColor.Highlight) : Color.FromArgb (0, 0, 128);
button_pressed_gradient_begin = use_system_colors ? Color.FromArgb (152, 181, 226) : Color.FromArgb (254, 128, 62);
button_pressed_gradient_end = use_system_colors ? Color.FromArgb (152, 181, 226) : Color.FromArgb (255, 223, 154);
button_pressed_gradient_middle = use_system_colors ? Color.FromArgb (152, 181, 226) : Color.FromArgb (255, 177, 109);
button_pressed_highlight = use_system_colors ? Color.FromArgb (150, 179, 225) : Color.FromArgb (150, 179, 225);
button_pressed_highlight_border = Color.FromKnownColor (KnownColor.Highlight);
button_selected_border = use_system_colors ? Color.FromKnownColor (KnownColor.Highlight) : Color.FromArgb (0, 0, 128);
button_selected_gradient_begin = use_system_colors ? Color.FromArgb (193, 210, 238) : Color.FromArgb (255, 255, 222);
button_selected_gradient_end = use_system_colors ? Color.FromArgb (193, 210, 238) : Color.FromArgb (255, 203, 136);
button_selected_gradient_middle = use_system_colors ? Color.FromArgb (193, 210, 238) : Color.FromArgb (255, 225, 172);
button_selected_highlight = use_system_colors ? Color.FromArgb (195, 211, 237) : Color.FromArgb (195, 211, 237);
button_selected_highlight_border = use_system_colors ? Color.FromKnownColor (KnownColor.Highlight) : Color.FromArgb (0, 0, 128);

check_background = use_system_colors ? Color.FromKnownColor (KnownColor.Highlight) : Color.FromArgb (255, 192, 111);
check_pressed_background = use_system_colors ? Color.FromArgb (152, 181, 226) : Color.FromArgb (254, 128, 62);
check_selected_background = use_system_colors ? Color.FromArgb (152, 181, 226) : Color.FromArgb (254, 128, 62);

grip_dark = use_system_colors ? Color.FromArgb (193, 190, 179) : Color.FromArgb (39, 65, 118);
grip_light = use_system_colors ? SystemColors.Window : Color.FromArgb (255, 255, 255);

image_margin_gradient_begin = use_system_colors ? Color.FromArgb (251, 250, 246) : Color.FromArgb (227, 239, 255);
image_margin_gradient_end = use_system_colors ? SystemColors.Control : Color.FromArgb (123, 164, 224);
image_margin_gradient_middle = use_system_colors ? Color.FromArgb (246, 244, 236) : Color.FromArgb (203, 225, 252);
image_margin_revealed_gradient_begin = use_system_colors ? Color.FromArgb (247, 246, 239) : Color.FromArgb (203, 221, 246);
image_margin_revealed_gradient_end = use_system_colors ? Color.FromArgb (238, 235, 220) : Color.FromArgb (114, 155, 215);
image_margin_revealed_gradient_middle = use_system_colors ? Color.FromArgb (242, 240, 228) : Color.FromArgb (161, 197, 249);
```

Зачем нужен статический анализ кода

```
button_checked_gradient_begin = use_system_colors ? Color.Empty : Color.FromArgb (255, 223, 154);
button_checked_gradient_end = use_system_colors ? Color.Empty : Color.FromArgb (255, 166, 76);
button_checked_gradient_middle = use_system_colors ? Color.Empty : Color.FromArgb (255, 195, 116);
button_checked_highlight = Color.FromArgb (195, 211, 237);
button_checked_highlight_border = Color.FromKnownColor (KnownColor.Highlight);
button_pressed_border = use_system_colors ? Color.FromKnownColor (KnownColor.Highlight) : Color.FromArgb (0, 0, 128);
button_pressed_gradient_begin = use_system_colors ? Color.FromArgb (152, 181, 226) : Color.FromArgb (254, 128, 62);
button_pressed_gradient_end = use_system_colors ? Color.FromArgb (152, 181, 226) : Color.FromArgb (255, 223, 154);
button_pressed_gradient_middle = use_system_colors ? Color.FromArgb (152, 181, 226) : Color.FromArgb (255, 177, 109);
button_pressed_highlight = use_system_colors ? Color.FromArgb (150, 179, 225) : Color.FromArgb (150, 179, 225);
button_pressed_highlight_border = Color.FromKnownColor (KnownColor.Highlight);
button_selected_border = use_system_colors ? Color.FromKnownColor (KnownColor.Highlight) : Color.FromArgb (0, 0, 128);
button_selected_gradient_begin = use_system_colors ? Color.FromArgb (193, 210, 238) : Color.FromArgb (255, 255, 222);
button_selected_gradient_end = use_system_colors ? Color.FromArgb (193, 210, 238) : Color.FromArgb (255, 203, 136);
button_selected_gradient_middle = use_system_colors ? Color.FromArgb (193, 210, 238) : Color.FromArgb (255, 225, 172);
button_selected_highlight = use_system_colors ? Color.FromArgb (195, 211, 237) : Color.FromArgb (195, 211, 237);
button_selected_highlight_border = use_system_colors ? Color.FromKnownColor (KnownColor.Highlight) : Color.FromArgb (0, 0, 128);

check_background = use_system_colors ? Color.FromKnownColor (KnownColor.Highlight) : Color.FromArgb (255, 192, 111);
check_pressed_background = use_system_colors ? Color.FromArgb (152, 181, 226) : Color.FromArgb (254, 128, 62);
check_selected_background = use_system_colors ? Color.FromArgb (152, 181, 226) : Color.FromArgb (254, 128, 62);

grip_dark = use_system_colors ? Color.FromArgb (193, 190, 179) : Color.FromArgb (39, 65, 118);
grip_light = use_system_colors ? SystemColors.Window : Color.FromArgb (255, 255, 255);

image_margin_gradient_begin = use_system_colors ? Color.FromArgb (251, 250, 246) : Color.FromArgb (227, 239, 255);
image_margin_gradient_end = use_system_colors ? SystemColors.Control : Color.FromArgb (123, 164, 224);
image_margin_gradient_middle = use_system_colors ? Color.FromArgb (246, 244, 236) : Color.FromArgb (203, 225, 252);
image_margin_revealed_gradient_begin = use_system_colors ? Color.FromArgb (247, 246, 239) : Color.FromArgb (203, 221, 246);
image_margin_revealed_gradient_end = use_system_colors ? Color.FromArgb (238, 235, 220) : Color.FromArgb (114, 155, 215);
image_margin_revealed_gradient_middle = use_system_colors ? Color.FromArgb (242, 240, 228) : Color.FromArgb (161, 197, 249);
```

Зачем нужен статический анализ кода

```
button_pressed_highlight = use_system_colors ?
                           Color.FromArgb (150, 179, 225) :
                           Color.FromArgb (150, 179, 225);
```

Mono

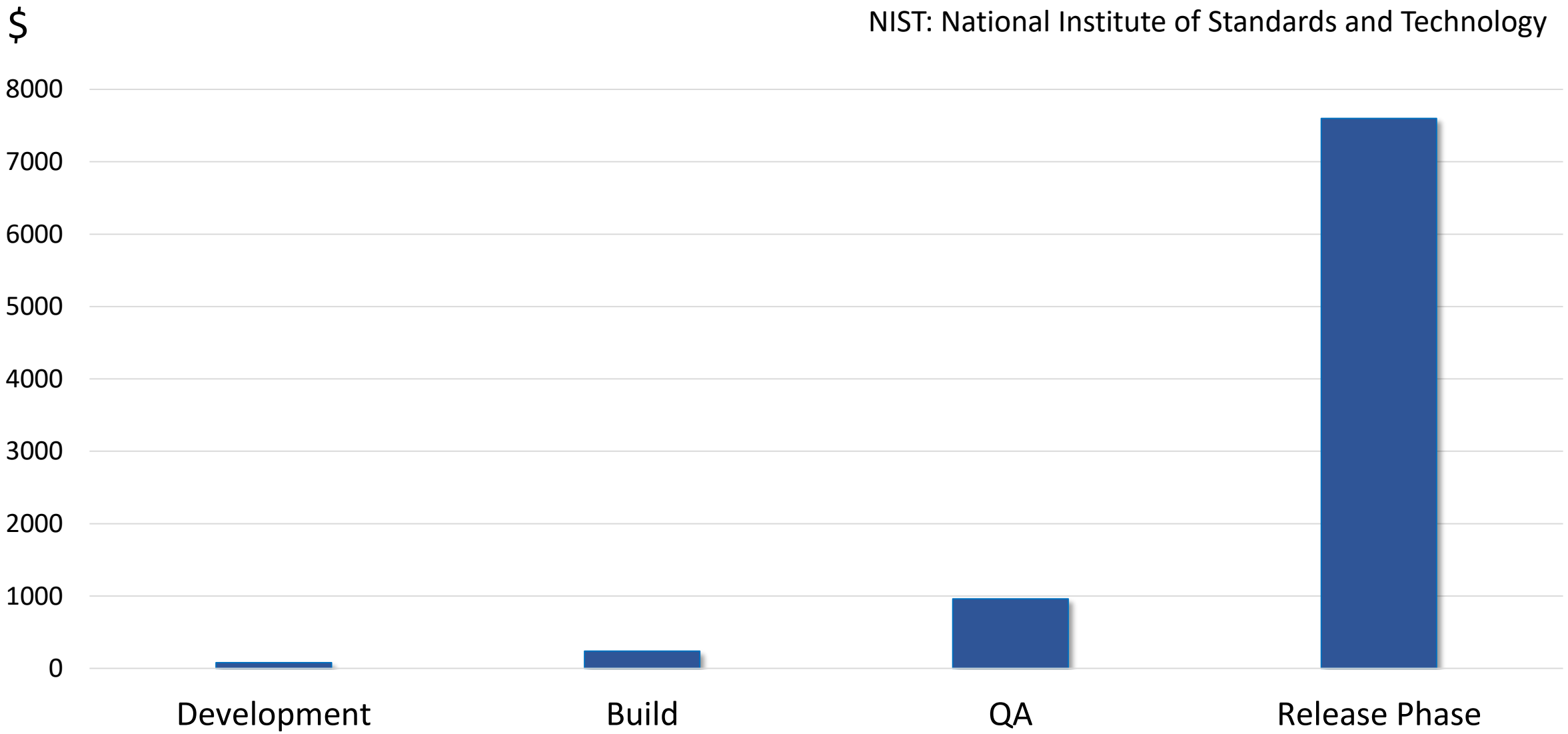
V3012 The '?:' operator, regardless of its conditional expression, always returns one and the same value: Color.FromArgb (150, 179, 225). ProfessionalColorTable.cs 258

Место статического анализа в процессе разработки

Только одно правило:

Чем ближе к коду – тем лучше

Стоимость исправления ошибки



Какие инструменты использовать

- PVS-Studio
- Coverity
- ReSharper
- SonarQube
- Klocwork
- Visual Studio
- Clang
- GCC
- …

Распространённые типы ошибок

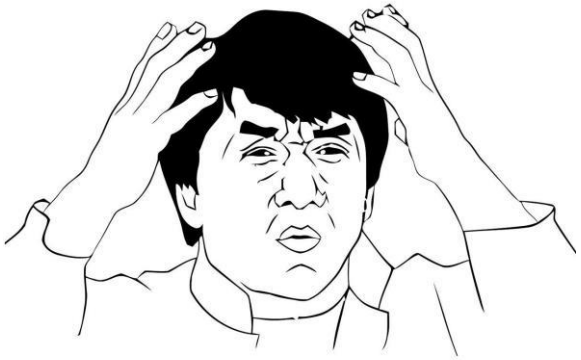
- Copy-Paste
- Доступ по нулевому указателю
- Ошибочные проверки
- Выход за границу массива
- Неиспользуемый результат
- Деление на ноль
- Некомпетентность
- …

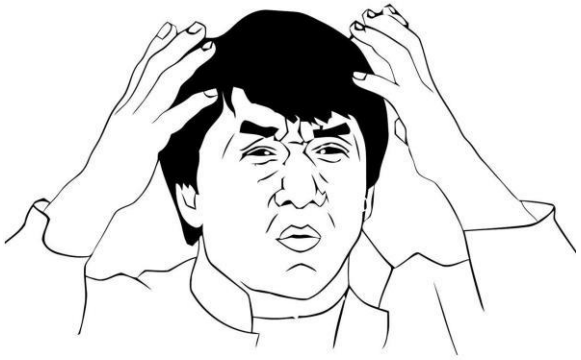
Roslyn

```
internal sealed class DynamicFileInfo
{
    public DynamicFileInfo(
        string filePath,
        SourceCodeKind sourceCodeKind,
        TextLoader textLoader,
        IDocumentServiceProvider documentServiceProvider)
    {
        FilePath = filePath;
        SourceCodeKind = SourceCodeKind;
        TextLoader = textLoader;
        DocumentServiceProvider = documentServiceProvider;
    }
```

Roslyn

```
internal sealed class DynamicFileInfo
{
    public DynamicFileInfo(
        string filePath,
        SourceCodeKind sourceCodeKind,
        TextLoader textLoader,
        IDocumentServiceProvider documentServiceProvider)
    {
        FilePath = filePath;
        SourceCodeKind = SourceCodeKind;
        TextLoader = textLoader;
        DocumentServiceProvider = documentServiceProvider;
    }
```





Amazon Web Services SDK для .NET

```
OnFailure? onFailure = null;

public OnFailure? OnFailure
{
   get { return  this.OnFailure; }
   set { this.onFailure = value; }
}
```

Amazon Web Services SDK для .NET

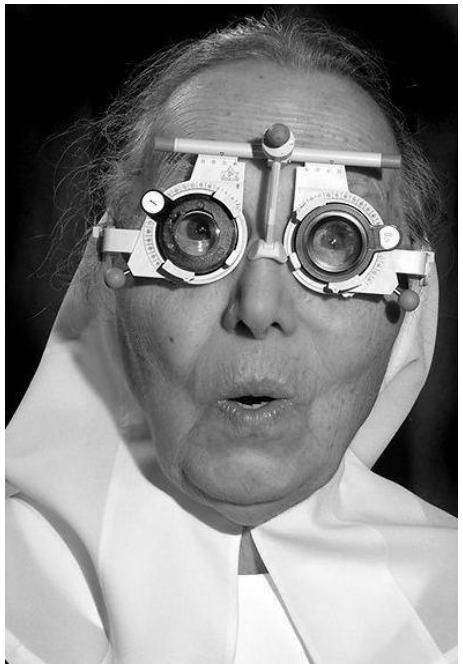
```
OnFailure? onFailure = null;

public OnFailure? OnFailure
{
   get { return  this.OnFailure; }
   set { this.onFailure = value; }
}
```

Unity

```
public override bool IsValid()
{
    ....
    return base.IsValid()
        && (pageSize >= 1 || pageSize <= 1000)
        && totalFilters <= 10;
}
```

Unity

```
public override bool IsValid()
{
    ....
    return base.IsValid()
        && (pageSize >= 1 || pageSize <= 1000)
        && totalFilters <= 10;
}
```



SAST - Static Application Security Testing

- Не только надёжный, но и безопасный код!
- Статический анализ, нацеленный на поиск и предотвращение уязвимостей
- Уязвимости - те же самые обыкновенные ошибки

Что за уязвимости?

CWE - Common Weakness Enumeration

CVE - Common Vulnerabilities and Exposures

Попробуй статический анализ

- Подходящий инструментарий
- Тонкая настройка
- Инкрементальный режим
- Регулярное использование

- Синтетические тесты
- Небольшая кодовая база

Внедрение

- Первый запуск гарантированно выдаст 100500 ошибок
- Рассматривай их как “технический долг”
- В дальнейшем эти ошибки можно и нужно спокойно исправить

Регулярное использование

- Настрой анализ на сборочном сервере во время ночных сборок

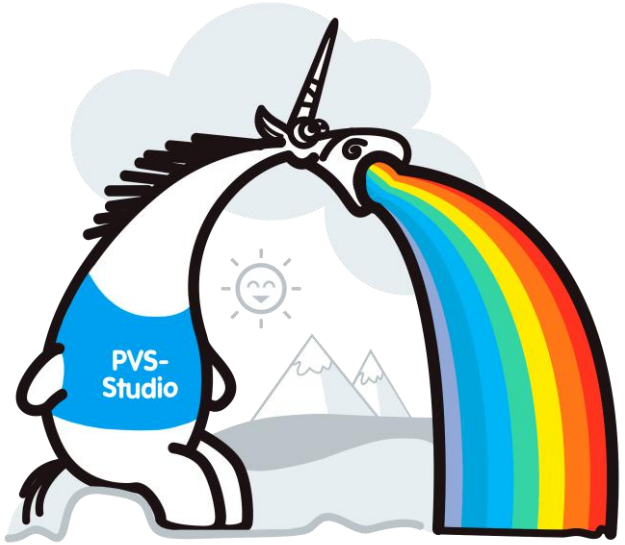
- Используй анализатор в инкрементальном режиме на машинах разработчиков

- Используй все другие доступные методики для улучшения качества кода

Выводы

- Статический анализ – не панацея от всех бед
- Статический анализ – это ответ на вопрос: "Что ещё я могу сделать для улучшения качества моего кода?"
- Что значит лучше? Легче поддерживать, проще развивать, быстрее устранять проблемы
- Если ваша компания зарабатывает деньги, используя программный код, вы просто не можете не использовать статический анализ кода

Ответы на вопросы



Контакты

Сергей Хренов

C# разработчик, PVS-Studio

khrenov@viva64.com

www.viva64.com