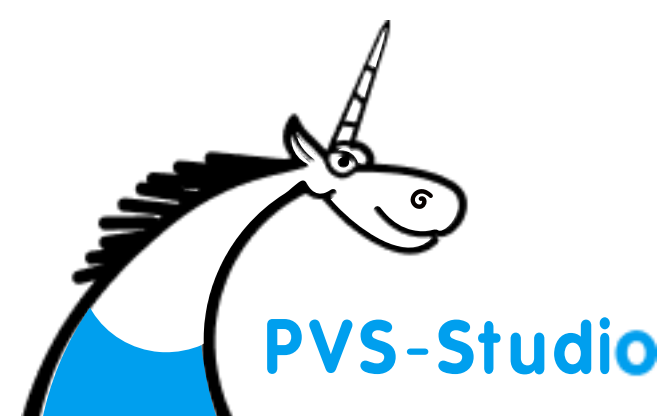
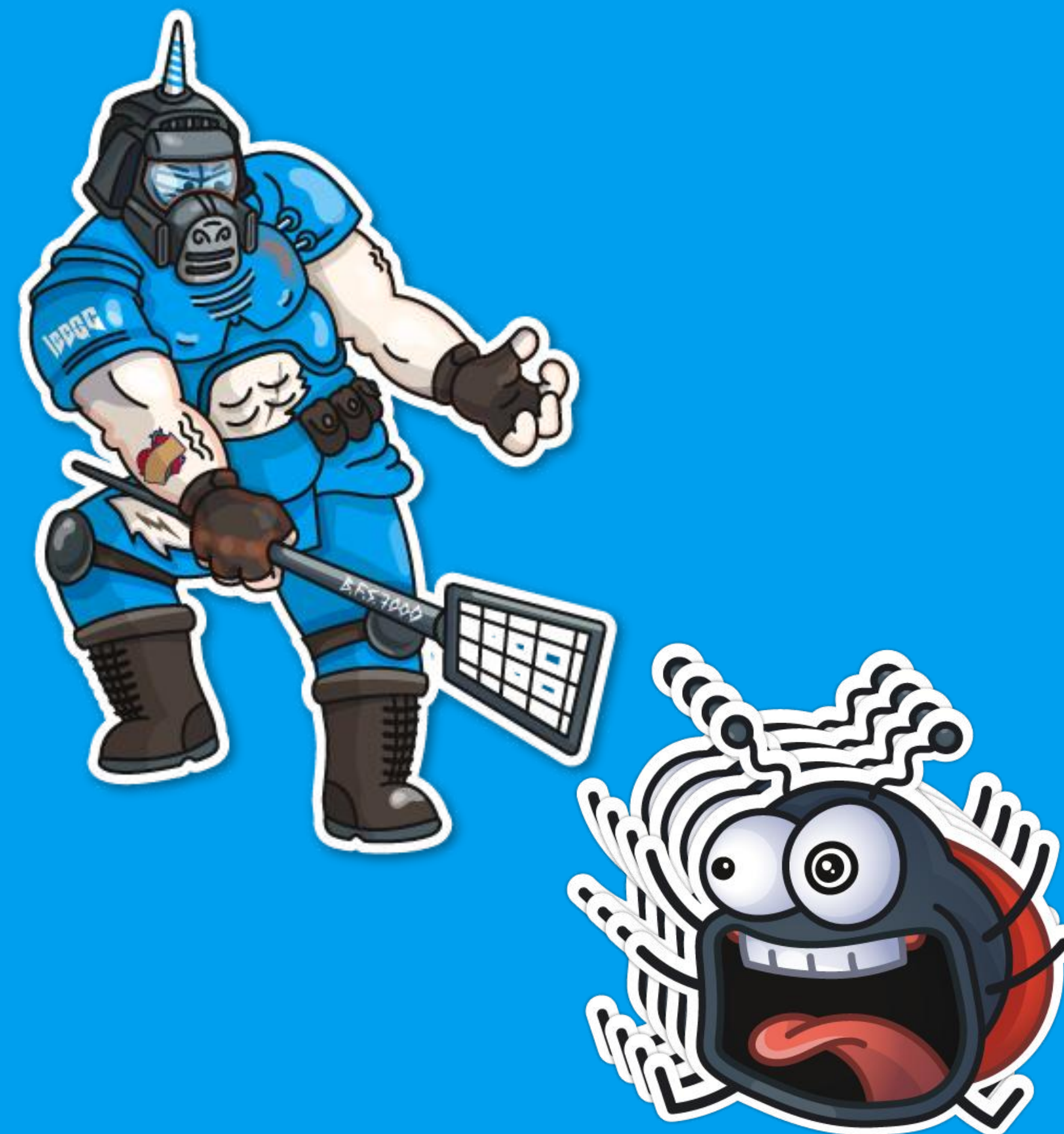




Вебинар

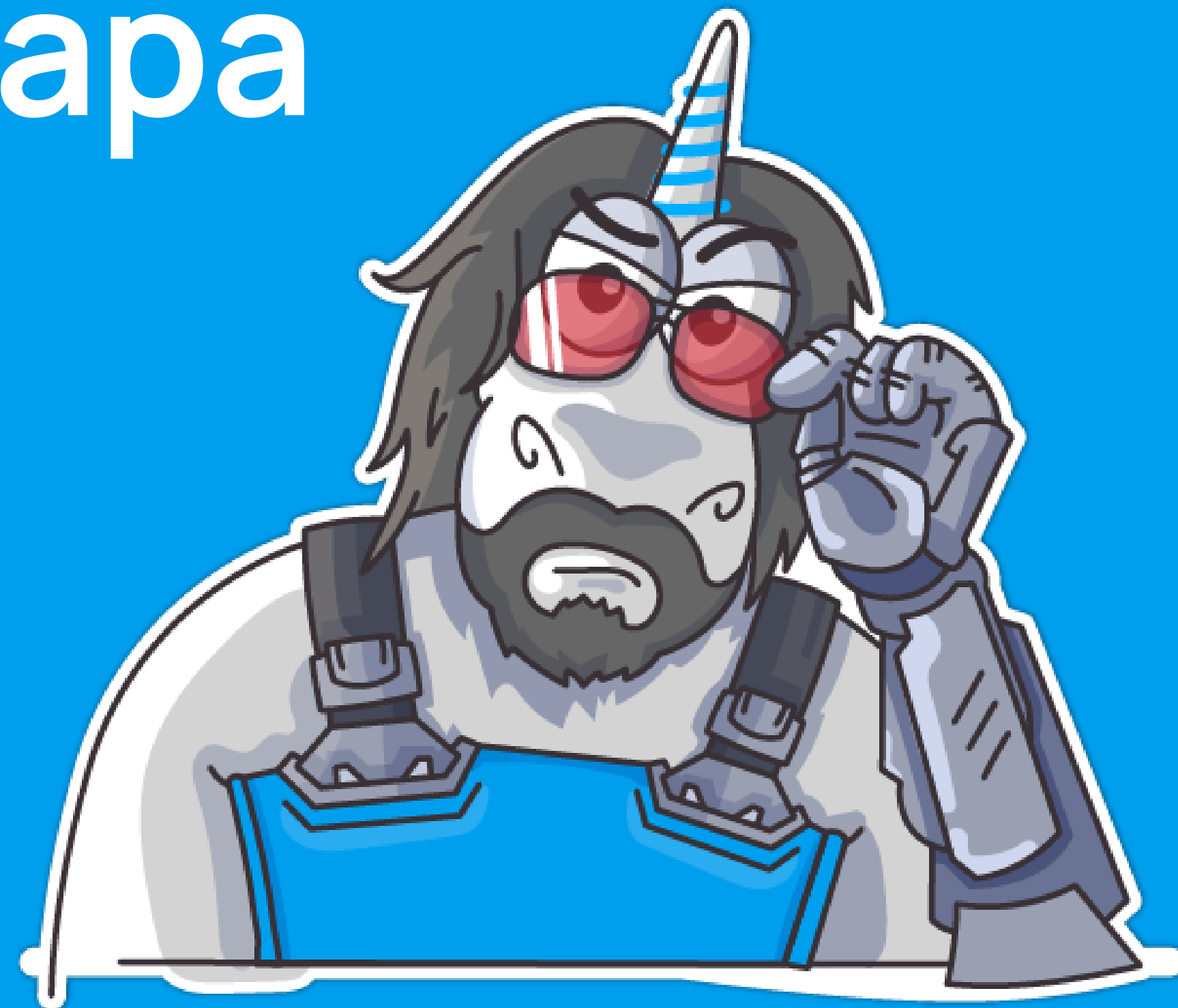
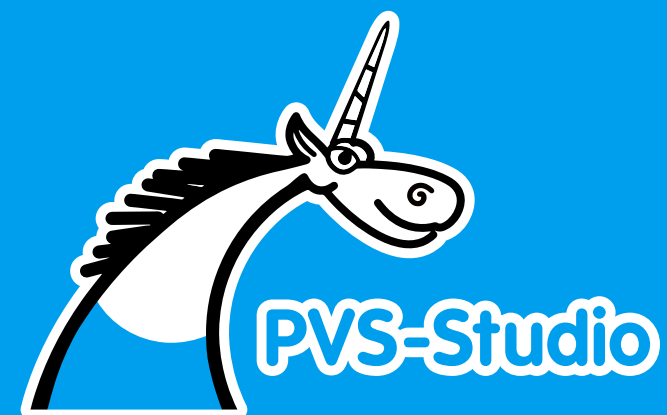
«Оптимизация игр»

работа со строками



4.12.2025

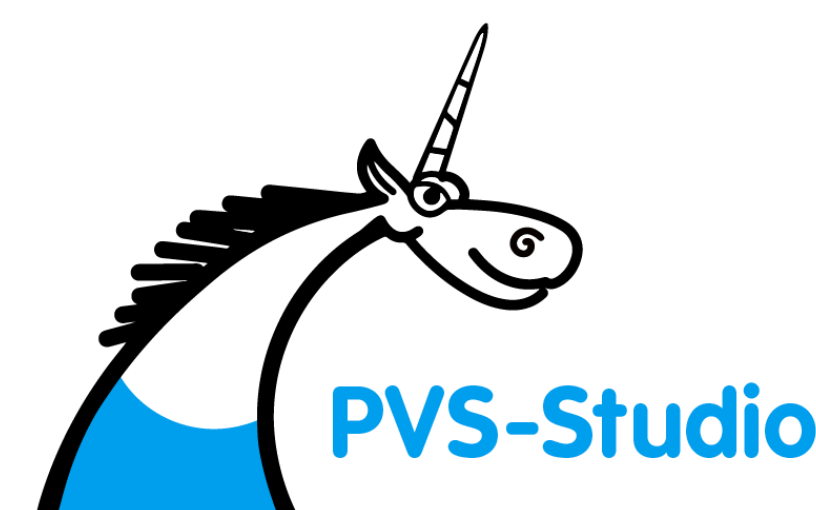
Спикеры вебинара



Глеб асламов

Developer Advocate, C# Developer

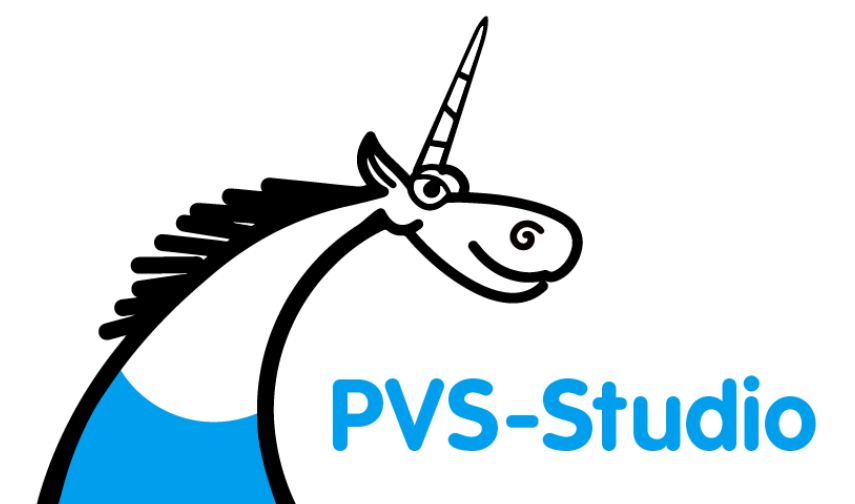
- Разработчик C# части анализатора PVS-Studio.
- Рассказываю про качество кода и безопасную разработку на конференциях, пишу технические и научные статьи для форумов и журналов.
- Модератор



Андрей Карпов

Директор по развитию бизнеса (CBDO)

- Один из основателей проекта PVS-Studio – <https://pvs-studio.ru/>
- 17 лет в сфере качества и анализа кода
- Хабр: [@Andrey2008](#)
- Email: [karpov \(@\) viva64.com](mailto:karpov (@) viva64.com)



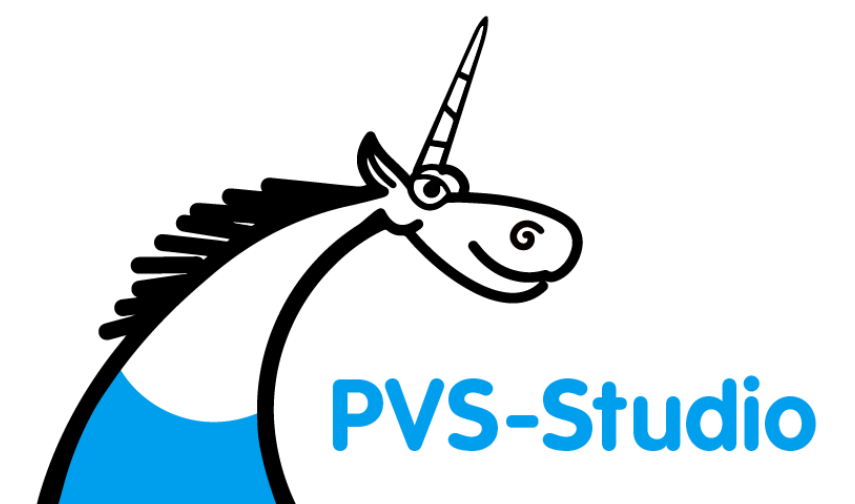
Андрей Карпов

Директор по развитию бизнеса (CBDO)

Доклад: «**std::string** – путешествие туда и обратно»

История о том, как в PVS-Studio мы создавали собственный класс строки и почему спустя годы отказались от него.

Расскажем о развитии стандартной библиотеки и внедрении Small String Optimization (SSO) в строки.



Денис Ярошевский

Performance engineer

- Активный член C++ сообщества
- Один из разработчиков библиотеки для векторизации EVE.



Денис Ярошевский

Performance engineer

Доклад: «Векторизованный поиск подстроки в строке»

Мы будем говорить о $n * m$ алгоритме поиска подстроки в строке. Несмотря на то что проблема кажется достаточно тривиальной, это не так: хорошая реализация будет значительно быстрее чем плохая.

Мы посмотрим на:

- * решения проблемы для собеседования
- * что на счет стандартной библиотеки
- * как устроено хорошее решение



Сергей Кушниренко

Senior Software Engineer в команде Age of Empires 2

- Занимается оптимизацией движка, крашами и гейзенбагами;
- Укладывает апдейт 6к объектов в 30мс в одном треде без внимания санитаров;
- В свободное время восстанавливает классический Pharaoh 1999 года в рамках проекта Akhenaten.



Сергей Кушниренко

Senior Software Engineer в команде Age of Empires 2

Доклад: «**String interning и все, все, все**»

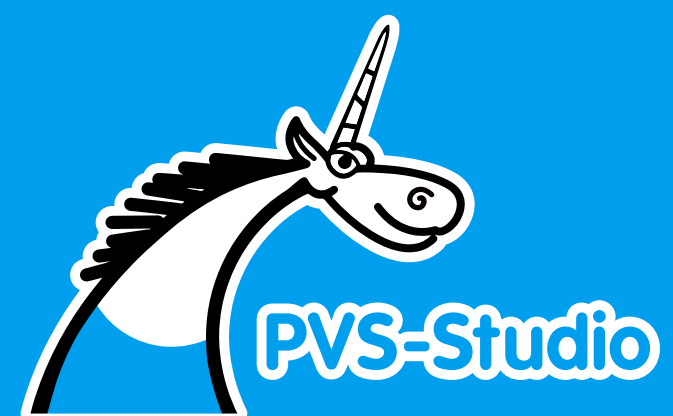
Пулы строк, xstring, идентификаторы,
SIMD-строки, immutable строки, StrHash
парадигма, строки-веревки:

рассмотрим, как можно экономить память
на текстовых данных и увеличить
скорость работы, приблизив её к
нативным операциям с числами.



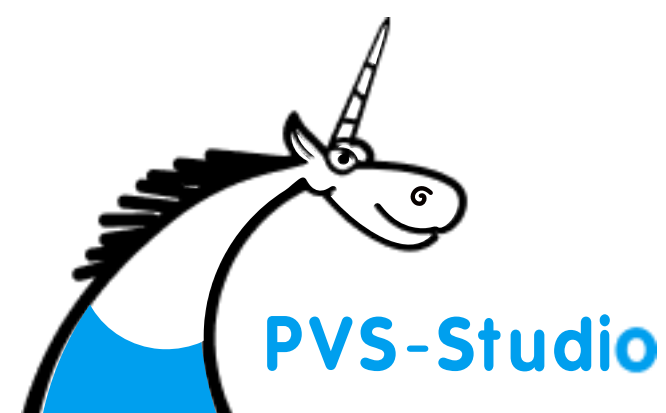


Переходим к докладам



А можно ли найти проблему в строках?

Мини-доклад



Глеб Асламов
C# Developer Advocate



Виды проблем

проблемы
безопасности

неправильная работа
с методами

недостижимый
код

ошибки доступа к
памяти

опечатки



ошибки сериализации /
десериализации

выход за
границы

ошибки
синхронизации

переполнение
буфера

неправильная работа с
типами

```
private void LateUpdate()
{
    ....
    if (ped != null)
        this.FocusPos = ped.transform.position;

    else if (Camera.main != null)
        this.FocusPos = Camera.main.transform.position;
    ....
    float relAngle = Camera.main != null ?
        Camera.main.transform.eulerAngles.y : 0f;
    ....
}
```



```
private void LateUpdate()  
{  
    ....  
    if (ped != null)  
        this.FocusPos = ped.transform.position;  
  
    else if (Camera.main != null)  
        this.FocusPos = Camera.main.transform.position;  
    ....  
    float relAngle = Camera.main != null ?
```

Предупреждение PVS-Studio:

- • • Предупреждение PVS-Studio: V4005 Expensive operation is performed inside the 'Camera.main' property. Using such property in performance-sensitive context can lead to decreased performance.

Code optimization

Considering performance in all the code you write helps your project scale without bottlenecks. There are several ways you can improve performance, including avoiding bad practices, [profiling](#) your code, implementing appropriate design patterns, and using techniques like asynchronous programming to split work across multiple threads of execution.

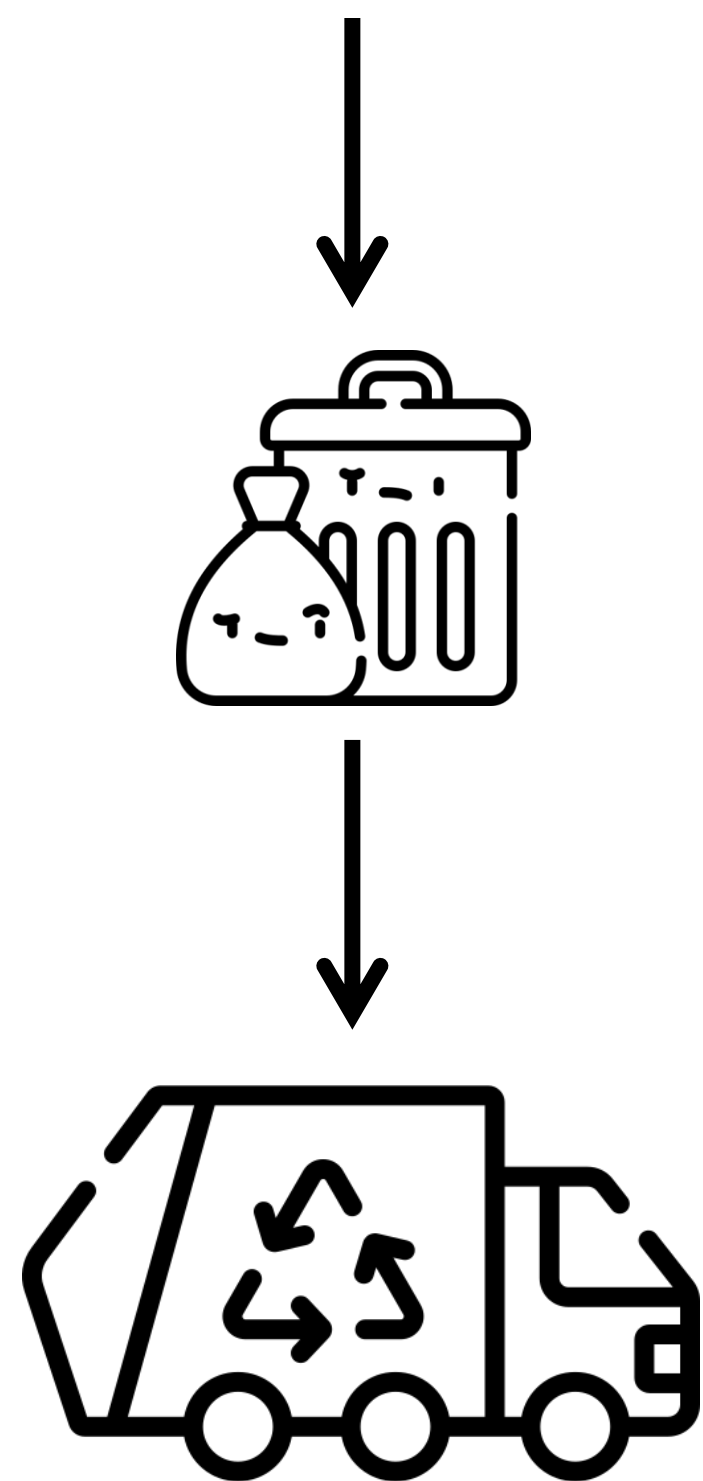
Topic	Description
Asynchronous programming	Asynchronous programming in Unity with the .NET <code>async</code> and <code>await</code> keywords and Unity's own custom <code>Awaitable</code> class.
Job system	Use Unity's own <code>Job</code> system to get the most out of multi-core CPUs and parallelize your algorithms.
Optimizing your code for managed memory	Approaches for optimizing your code to work with managed memory.

Проблема конкатенации

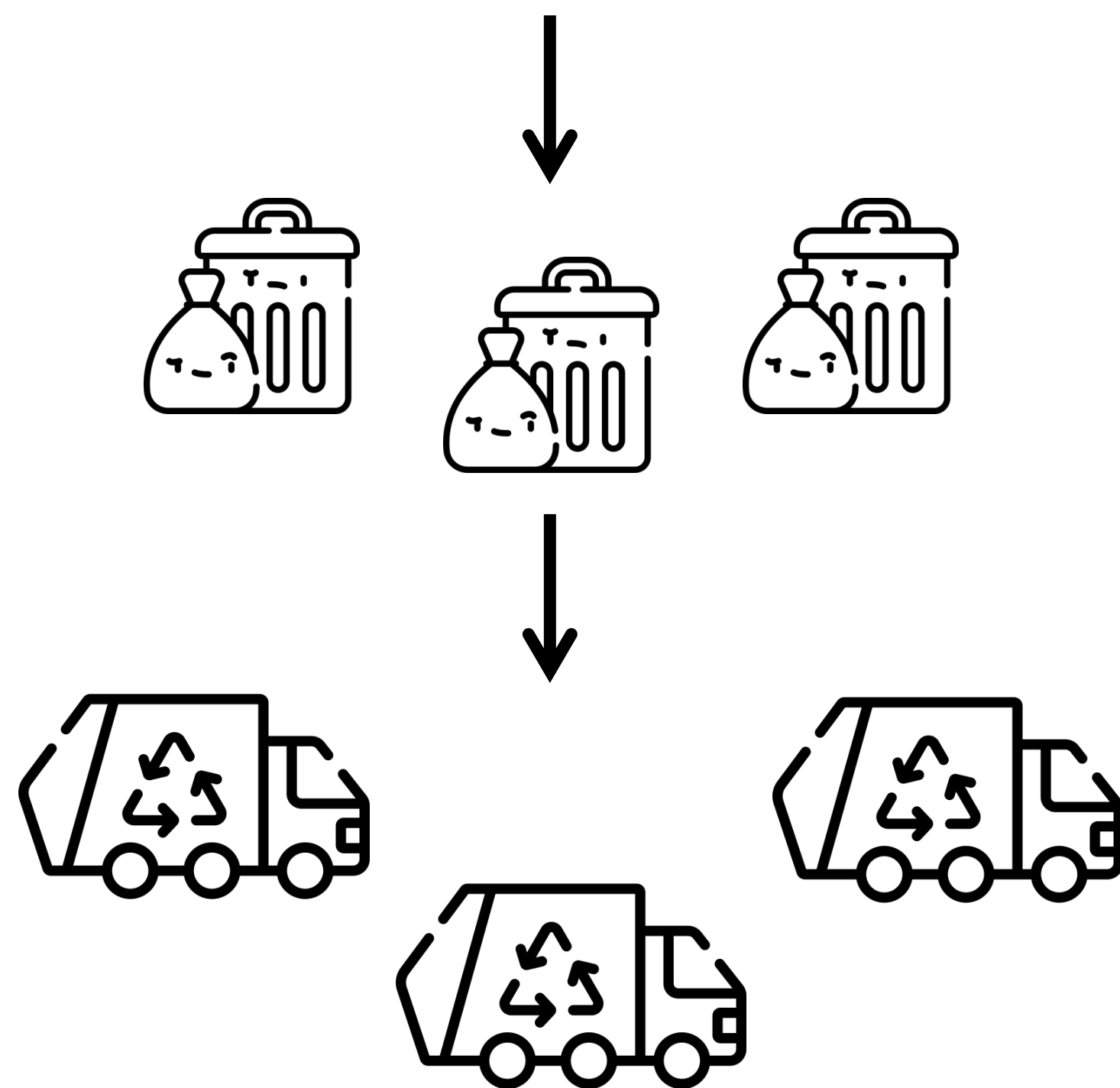
(~60 раз в секунду)

17

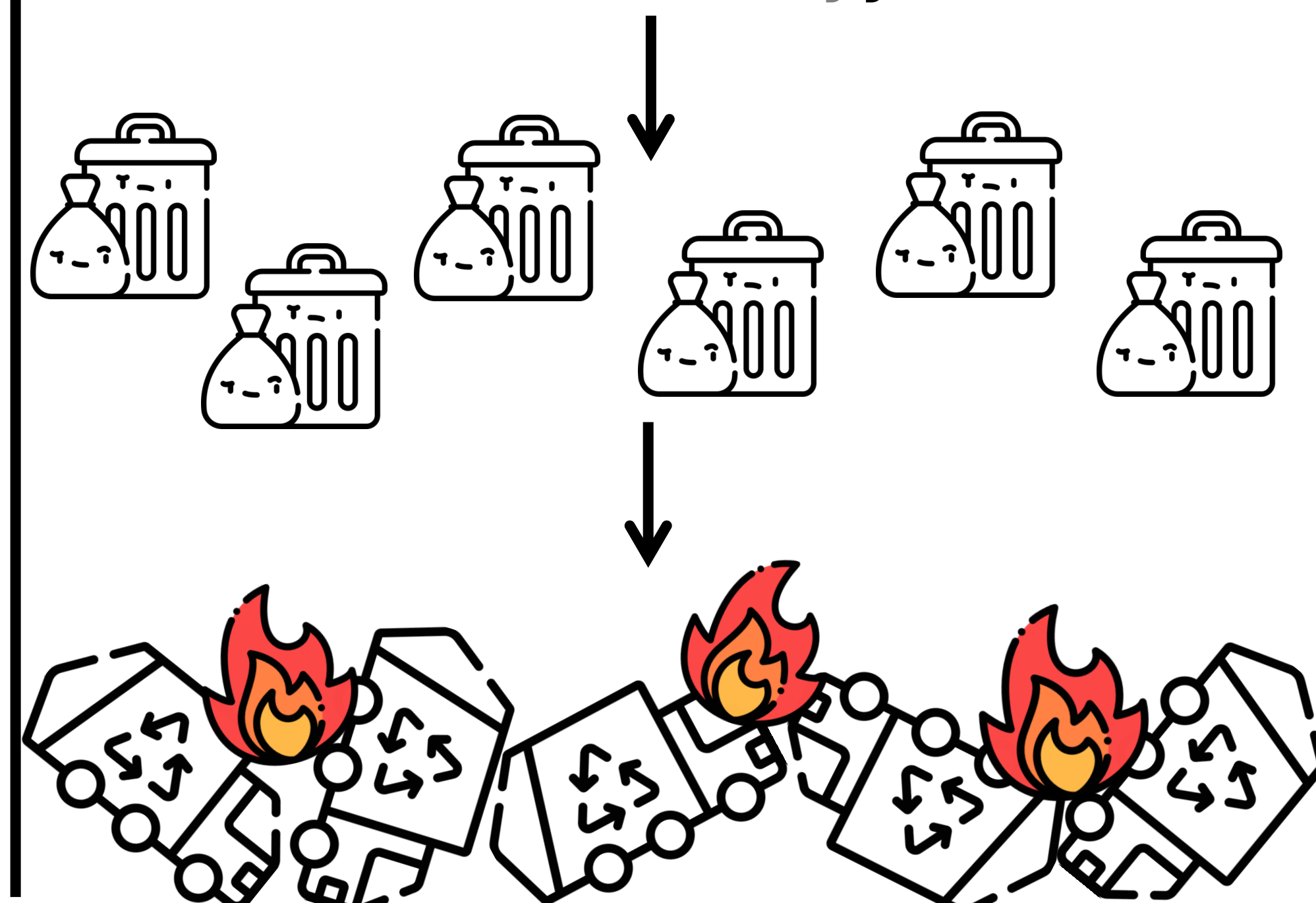
```
string str = "hello";  
str += " ";  
str += "world";
```



```
for(int i = 0; i < 100; i++) {  
    string str = "hello";  
    str += " ";  
    str += "world";  
}
```



```
void Update() {  
    for(int i = 0; i < 100; i++) {  
        string str = "hello";  
        str += " ";  
        str += "world";  
    }  
}
```



Алгоритм работы [1 часть]

18

```
void Update() 1 {  
    string message = null;  
    for(int i = 0; i < 100; i++) {  
        2 if(message == null or i > 50) {  
            message += "hello";  
        }  
    }  
    3  
    message += "worlds";  
    ...  
}
```

Этапы

- 1) Посещение метода
- 2) Сбор статистики
- 3) Поиск события

Алгоритм работы [2 часть]

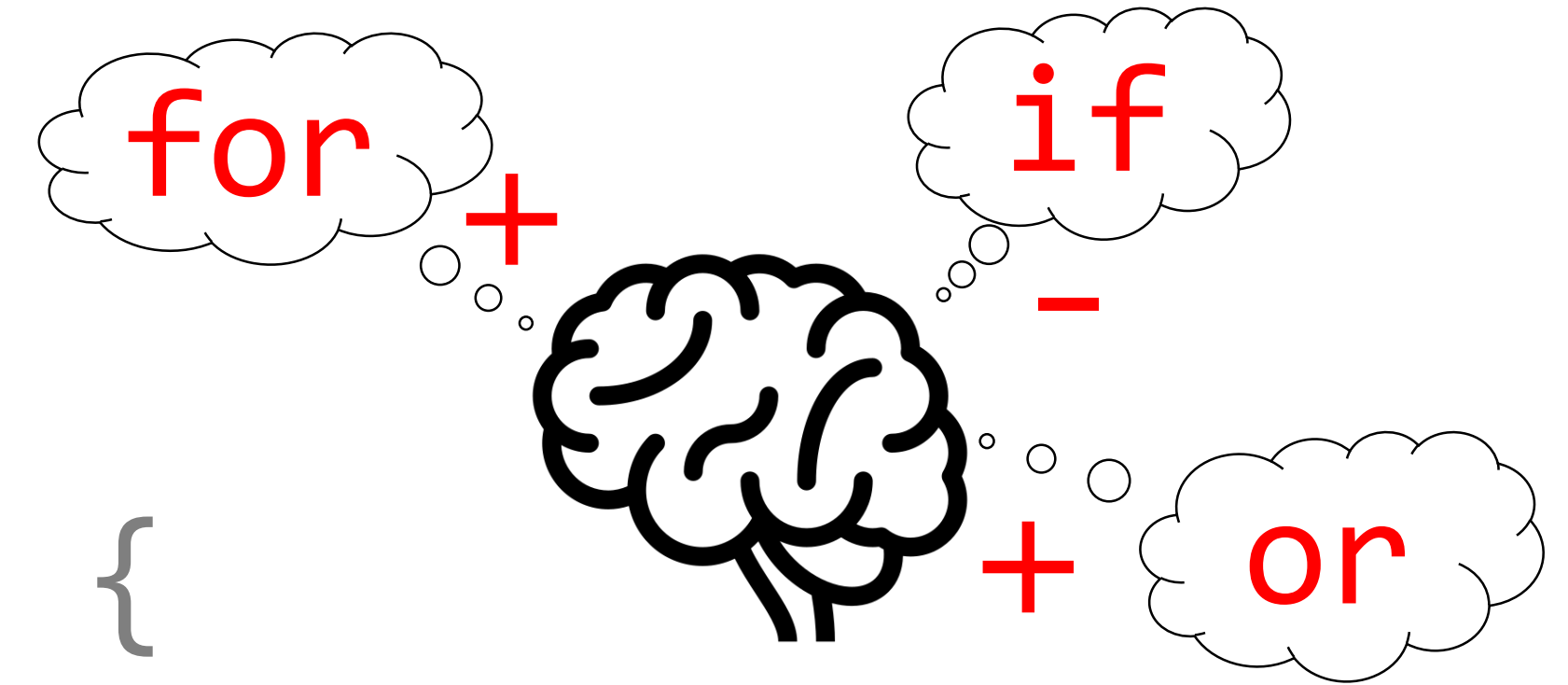
19

```
void Update() {  
    string message = null;  
    for(int i = 0; i < 100; i++) {  
        if(message == null or i > 50) {  
            message += "hello";  
        }  
    }  
}
```

```
message += "worlds";
```

4

Калькулятор



5

6

Этапы

- 4) Расчет частоты
- 5) Формирование срабатывания
- 6) Выдача предупреждения

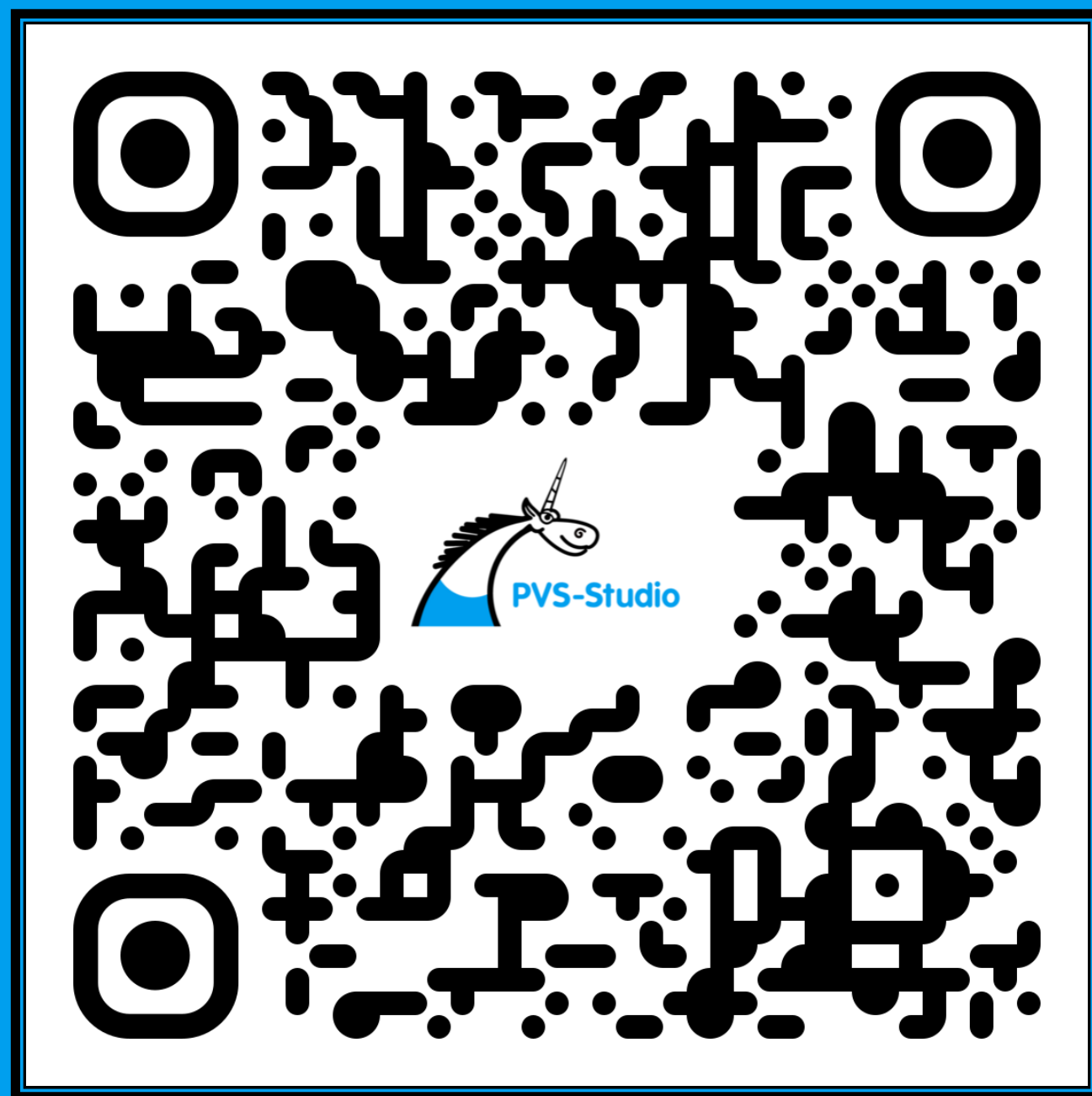
V4002. Avoid storing consecutive concatenations inside a single string in performance-sensitive context.

```
[SerializeField] Text _stateText;
....
void Update()
{
    ....
    string stateInfo = ....;
    ....
    stateInfo += ....;
    stateInfo += ....;
    ....
    stateInfo += ....;
    _stateText.text = stateInfo;
    ....
}
```

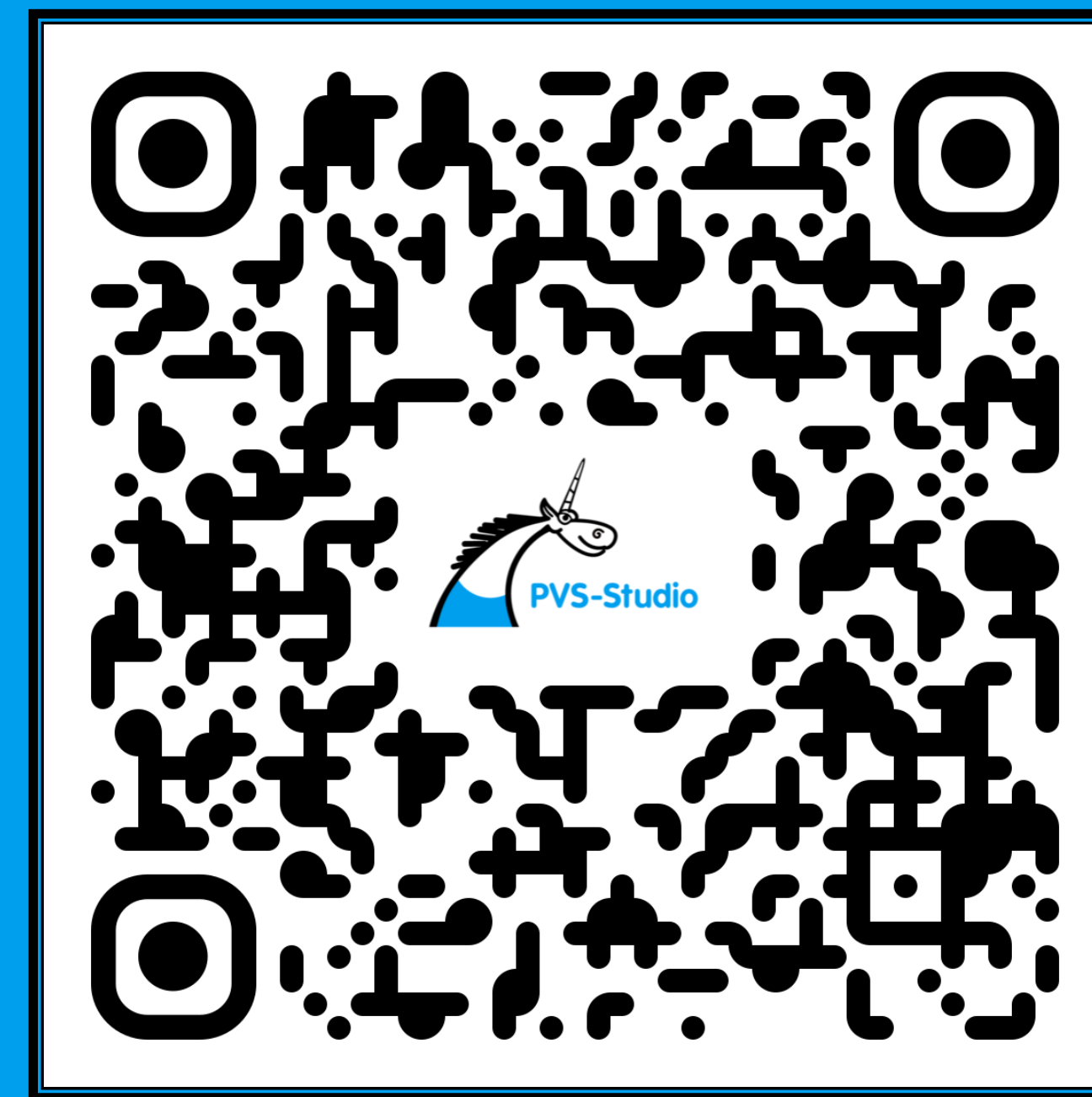


```
[SerializeField] Text _stateText;
....
StringBuilder _stateInfo = new StringBuilder();
void Update()
{
    _stateInfo.Clear();
    ....
    _stateInfo.AppendLine(...);
    _stateInfo.AppendLine(...);
    ....
    _stateInfo.AppendLine(...);
    _stateText.text = _stateInfo.ToString();
    ....
}
```

Материалы по теме

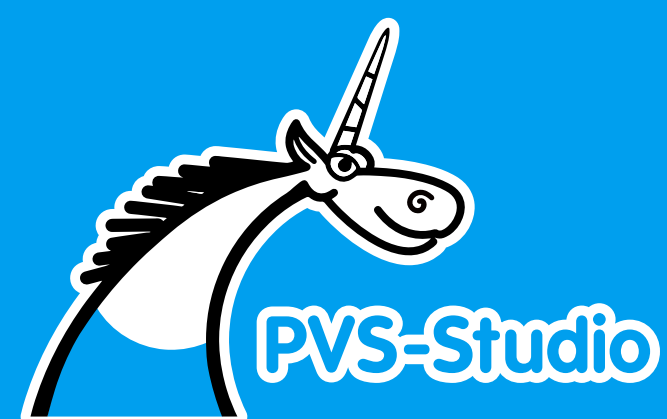
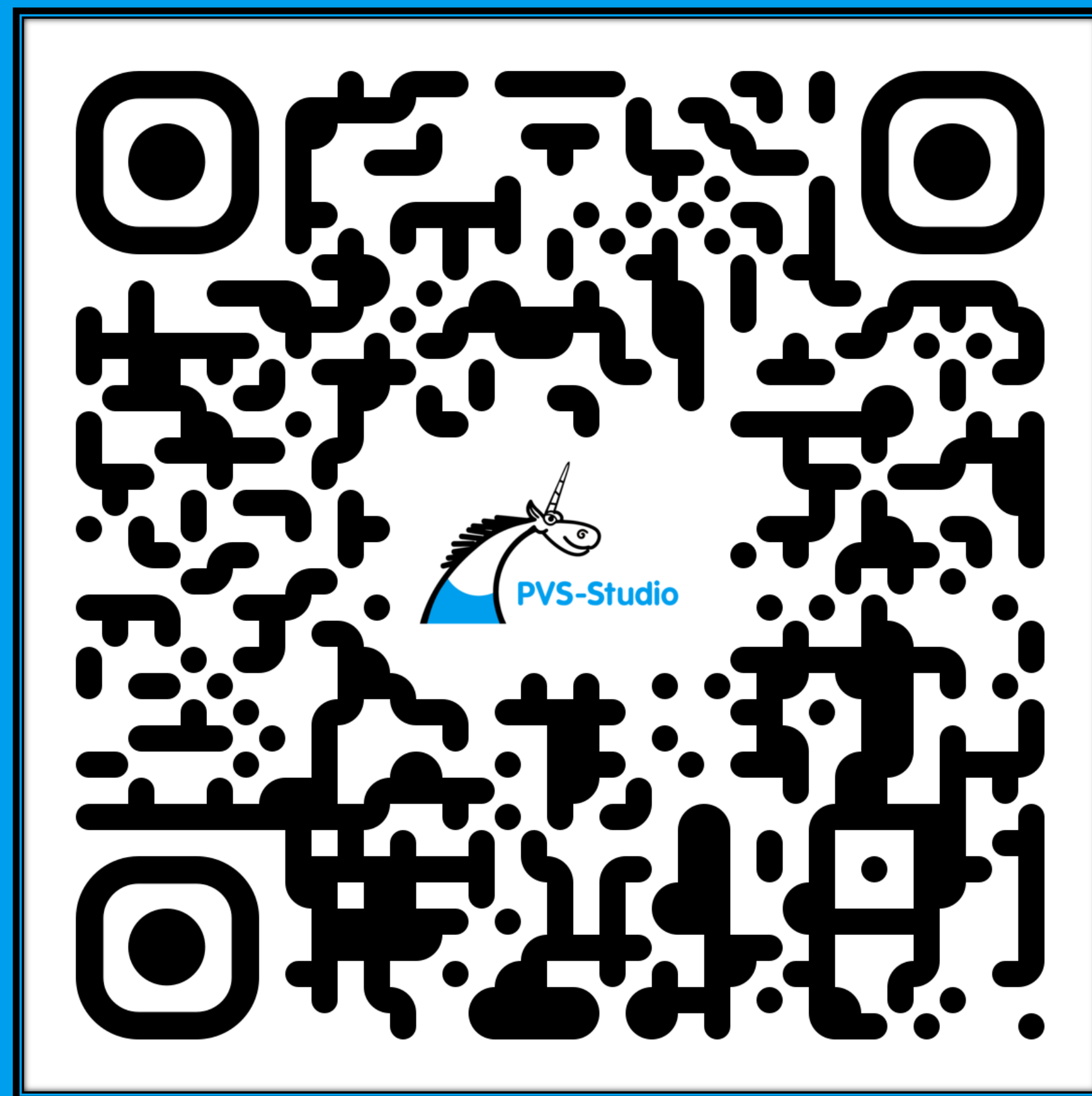


На защите GameDev'а:
статический анализ и Unity



Меньше багов — больше FPS: как
статический анализ помогает
проектам на Unreal Engine

Помогите нам стать лучше!



pvs-studio.ru/ru/about-feedback/

Сделай свой проект чистым и
безопасным вместе
с PVS-Studio



Курс «Нескучное
программирование.
C++ без аллокаций памяти»



Слайды к SIMD
Substring in a String

