IOA Error Panel

MSK.NET представляет

Обработка ошибок в C#

PLEASE PRESS ENTER TO RESUME

ERROR

Илья Фофанов

ENGINEER SPOCK

http://engineerspock.com

Зачем надо уметь обрабатывать ошибки?

- Бич современности: падающие приложения, которые ничего не сообщают о возникшей проблеме
- Обработка ошибок – критически важная часть программ с высокими требованиями к устойчивости


Проверяемые
VS
Непроверяемые


Большие проблемы…
с исключениями

Виды ошибок

Как бороться с исключениями

«Try-Catch» Hell

```
public HttpResponseMessage CreateCustomer(string name,
                                          string billingInfo) {
    Result<BillingInfo> billingInfoResult = null;
    try {
        billingInfoResult = BillingInfo.Create(billingInfo);
    }
    catch (Exception ex) {
        Log(ex);
        return CreateResponseMessage(ex);
    }

    Result<CustomerName> customerNameResult = null;
    try {
        customerNameResult = CustomerName.Create(name);
    }
    catch (Exception ex) {
        Log(ex);
        return CreateResponseMessage(ex);
    }

    try {
        _paymentGateway.ChargeCommission(billingInfoResult.Value);
    }
    catch (Exception ex) {
        Log(ex);
        return CreateResponseMessage(ex);
    }

    var customer = new Customer(customerNameResult.Value);
    try {
        _repository.Save(customer);
    }
    catch (Exception ex) {
        Log(ex);
        _paymentGateway.RollbackLastTransaction();
    }
    try {
        _emailSender.SendGreetings(customerNameResult.Value);
    }
    catch (Exception ex) {
        Log(ex);
        return CreateResponseMessage(ex);
    }
    return CreateResponseMessage(true);
}
```

«No Try-Catch» Paradise

```
public HttpResponseMessage CreateCustomer(string name, string billingInfo)
{
    Result<BillingInfo> billingInfoResult = BillingInfo.Create(billingInfo);
    Result<CustomerName> customerNameResult = CustomerName.Create(name);

    return Result.Combine(billingInfoResult, customerNameResult)
        .OnSuccess(() => _paymentGateway.ChargeCommission(billingInfoResult.Value))
        .OnSuccess(() => new Customer(customerNameResult.Value))
        .OnSuccess(customer => _repository.Save(customer)
                                          .OnFailure(() => _paymentGateway.RollbackLastTransaction()))
        .OnSuccess(() => _emailSender.SendGreetings(customerNameResult.Value))
        .OnBoth(Log)
        .OnBoth(CreateResponseMessage);
}
```

Семь бед - один ответ: **исключения!**

“

Use exceptions. They are much cleaner then all the other options.
The fear about uncaught exceptions is misplaced. Forgetting to catch an exception is much better than forgetting the if statement to check a result. The former fails visibly, and the latter fails silently.
The fear of messy code near the try/catch blocks is also misplaced. Use the 'extract till you drop' rule, and make sure that any function with a try/catch block has _only_ the try/catch block in it; with the try block being a single function call.
Finally, write tests for all your exception throws, and all your exception catches. Error processing is part of the legitimate behavior of your system and testing it is very important.

Robert Martin aka “Uncle Bob”

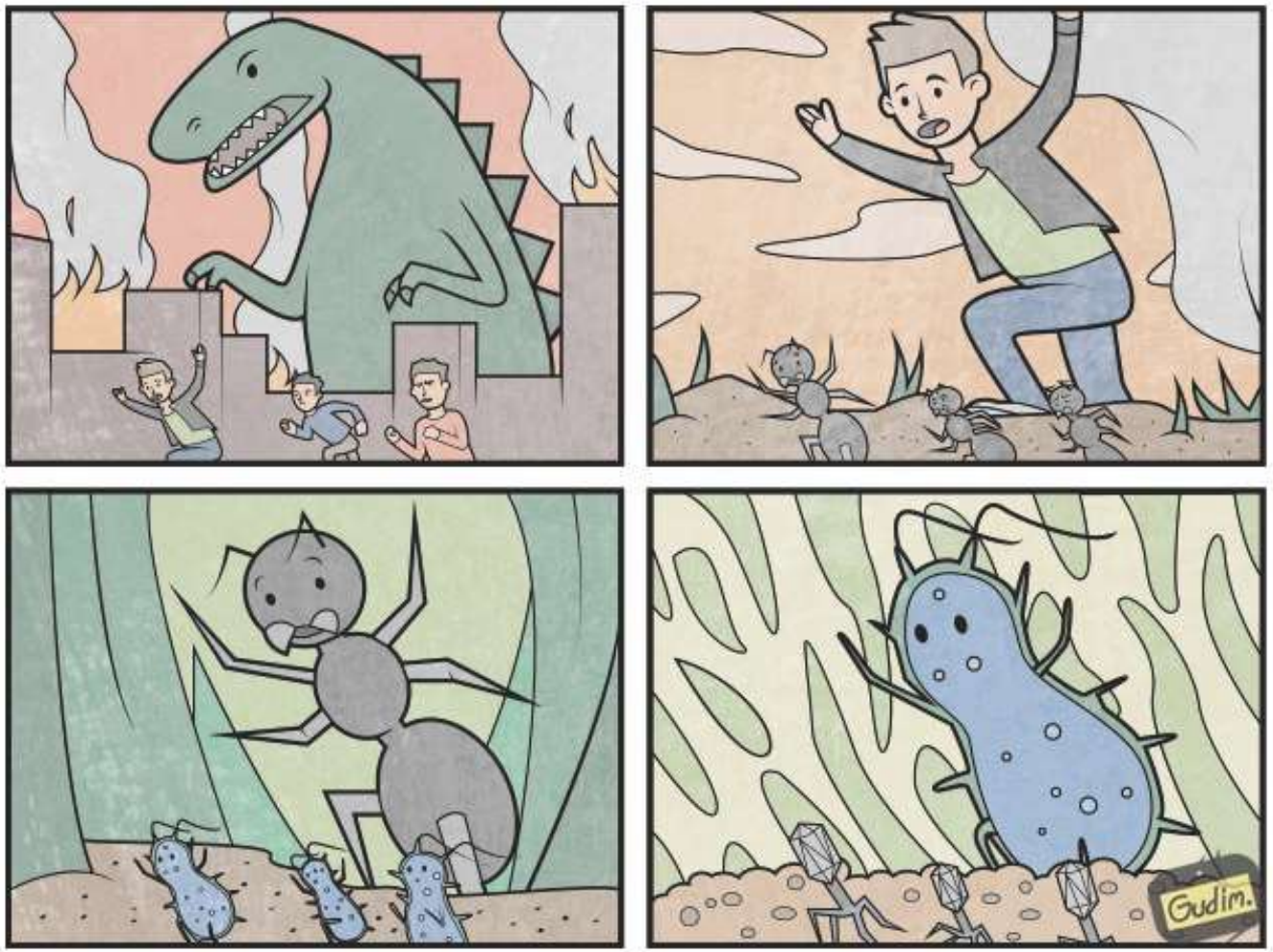
https://groups.google.com/forum/#!topic/clean-code-discussion/CQSA4czDvHc

Проблемы исключений

Пример проверяемого исключения

```
public static void readFile(string filePath) throws IOException {
    FileReader file = new FileReader(filePath);
    BufferedReader fileInput = new BufferedReader(file);

    // Print first 3 lines of the file
    for (int counter = 0; counter < 3; counter++)
        System.out.println(fileInput.readLine());

    fileInput.close();
}
```

Проблема масштабирования



Проблема версионирования

Нельзя просто так взять

и добавить новый тип исключения

Какое решение?

Выпилить проверяемые исключения!

Трудно понять программу

“

Their power comes at an extremely high cost; it becomes impossible to understand the flow of the program using only local analysis; the whole program must be understood. This is especially true when you mix exceptions with more exotic control flows like event-driven or asynchronous programming. Avoid, avoid, avoid; use exceptions only in the most exceptional circumstances, where the benefits outweigh the costs.

Eric Lippert

“

“…people don't care. They're not going to handle any of these exceptions. **There's a bottom level exception handler around their message loop.** That handler is just going to bring up a dialog that says what went wrong and continue.”

“It is funny how people think that the important thing about exceptions is handling them. That is not the important thing about exceptions. In a well-written application **there's a ratio of ten to one, in my opinion, of try finally to try catch**.”

Anders Hejlsberg

Как обезопасить себя при чтении файла?

```
try
{
    File.ReadAllLines();
}
catch (?)
{

}
```

Исключения из File.ReadAllLines()

- ArgumentException
- ArgumentNullException
- PathTooLongException
- NotSupportedException

- UnathorizedAccessException
- SecurityException

- IOException
- DirectoryNotFoundException
- FileNotFoundException

Знаем ли мы какие исключения перехватывать?

Corrupted State Exceptions

SEHException

Если CLR не знает, что делать с исключением из unmanaged кода, то CLR заворачивает его в тип SEHException.

Corrupted State Exception

Corrupted State Exception (CSE) –
исключение связанное с поврежденным состоянием.

Примеры CSE в C#:

- SEHException
- AccessViolationException

CSE в unmanaged:

- EXCEPTION_ILLEGAL_INSTRUCTION EXCEPTION_IN_PAGE_ERROR
- EXCEPTION_INVALID_DISPOSITION
  EXCEPTION_NONCONTINUABLE_EXCEPTION
- EXCEPTION_ACCESS_VIOLATION EXCEPTION_STACK_OVERFLOW
- EXCEPTION_PRIV_INSTRUCTION STATUS_UNWIND_CONSOLIDATE

CSE. Надо ли ловить?

- **В общем случае ловить CSE не надо. Компенсационную логику написать практически невозможно.**

- **Можно ловить только в случаях, когда вы точно знаете с какой проблемой столкнулись и уверены, что можно «подавить и продолжить»**

CSE. Как поймать?

1.

```
[HandleProcessCorruptedStateExceptions]
public static void HandleCorruptedState()
{
    // handling
}
```

2.

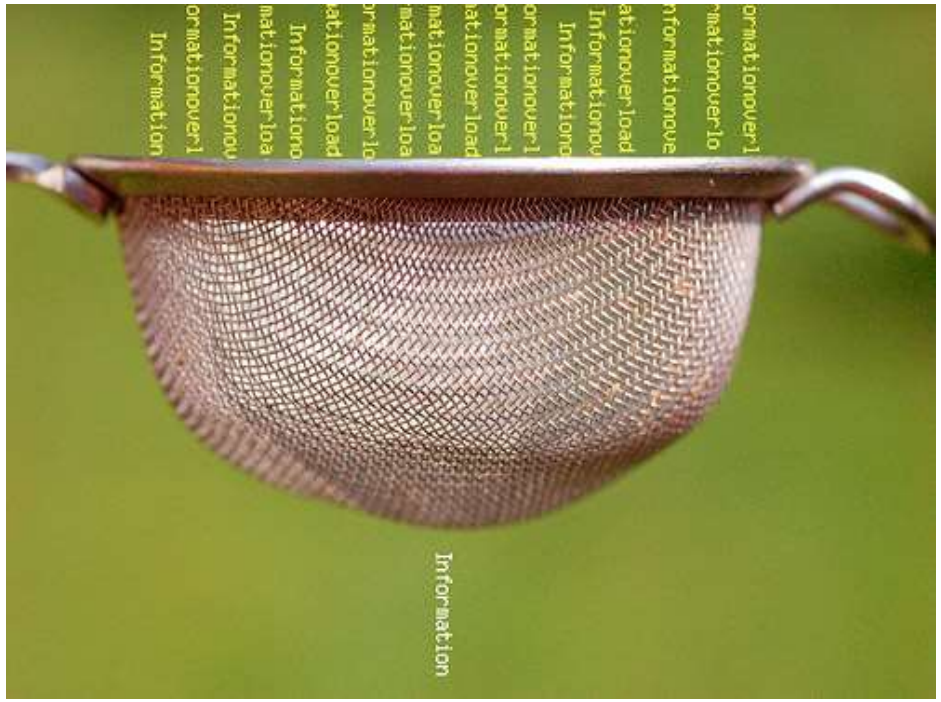
```
<configuration>
    <runtime>
        <legacyCorruptedStateExceptionsPolicy enabled="true"/>
    </runtime>
</configuration>
```

Обработка исключений

Обработка только известных исключений

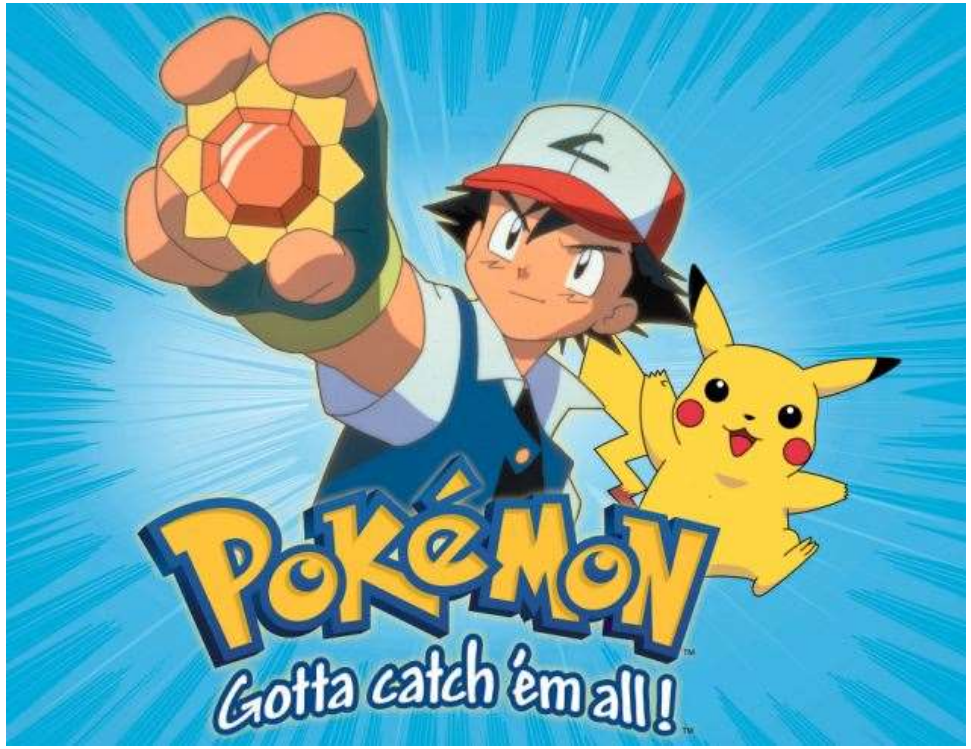
```
try {
    File.ReadAllLines();
}
catch (IOException ex) {

}
//а точно ничего не забыли?
```

Политики фильтраций




```
try {
    File.ReadAllLines();
}
catch (Exception ex) {
    //делегируем обработку
    ExceptionManager
      .HandleException(ex, “Policy”);
}
```

Catch Them All!




```
try {
    File.ReadAllLines();
}
//вообще всё пофигу!
catch (Exception ex) {

}
```

Подавление исключений

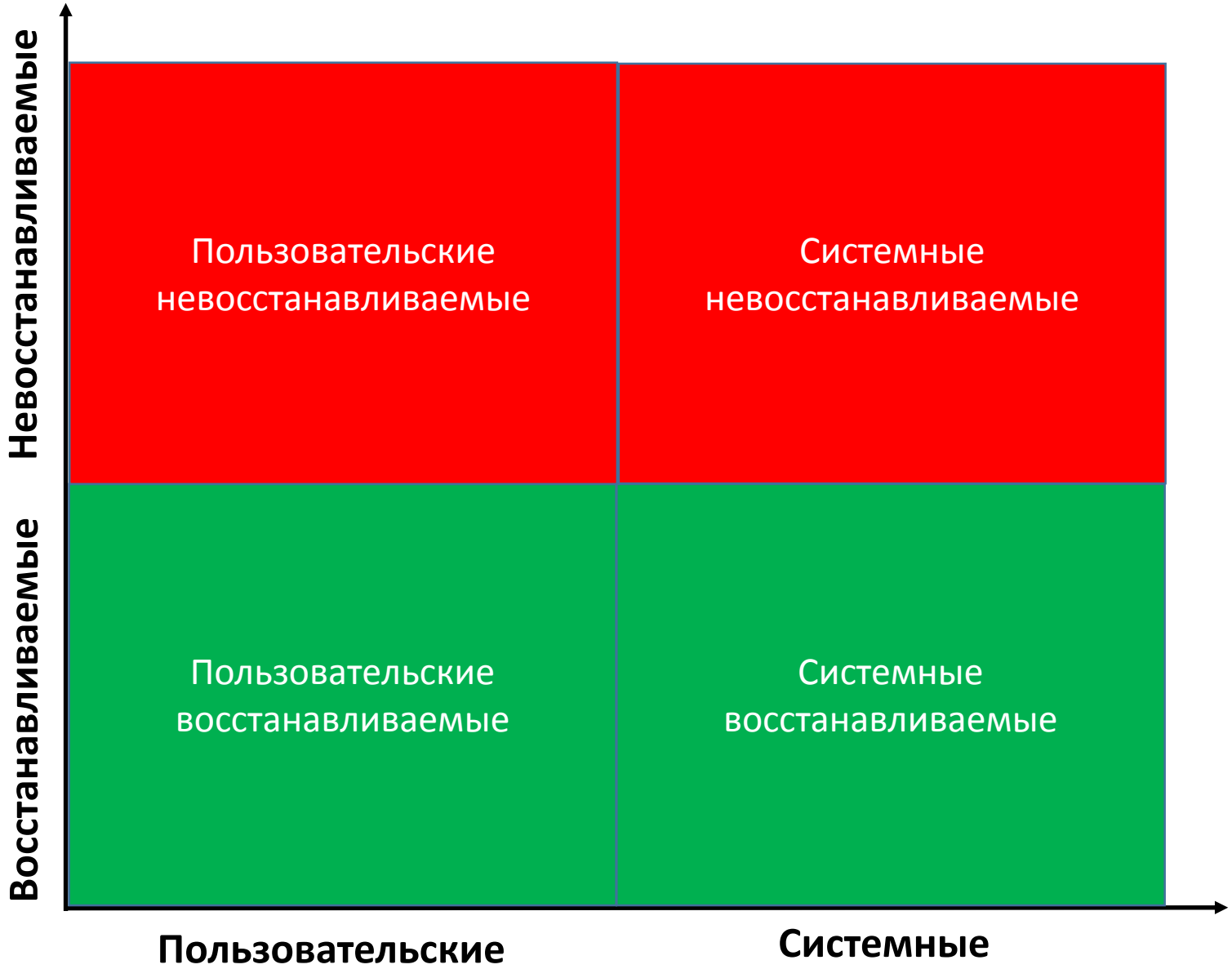
1. Приложение какого типа вы разрабатываете?
2. Полной корректности достичь очень трудно.
3. Каков размер приложения?

“

… most people don’t write robust error handling code in non-systems programs, throwing an exception *usually* gets you out of a pickle fast. Catching and then proceeding often works too. No harm, no foul. Statistically speaking, programs “work.”

Joe Duffy

Категории ошибок



Восстанавливаемые пользовательские ошибки

```
public void TransferMoney(Payment p)
{
    if (p == null)
        throw new ArgumentNullException("p");
    ValidatePayment(p);
}
```

Исключения и сигнатура метода

- Исключения – побочные эффекты
- Исключения, выражающие нарушения пользовательской бизнес-логики скрывают важную семантическую часть

Эту семантическую часть надо вывести в сигнатуру метода!

Боремся с исключениями

Способы борьбы с исключениями

- Шаблон Tester-Doer
- Шаблон TryParse
- Railway Oriented Programming

Tester-Doer

```
ICollection<Student> students = GetStudents();
if (!students.IsReadOnly)//tester
{
   students.Add(new Student("Joe")); //doer
}
```

TryParse

```
DateTime birthday;
bool parsed = DateTime.TryParse("12/08/1988", out birthday);
if (parsed)
{
    SocialNumber socialNumber;
    parsed = SocialNumber.TryParse("1239584594", out socialNumber);
    if (parsed) { }
    else { }
}
else
{
}
```

Недостатки Tester-Doer и TryParse

- Tester-Doer – форма антипаттерна «temporal coupling»
- Tester-Doer не работает в условиях конкурентного доступа
- TryParse неуклюж из-за out-параметра и возврата boolean

Pipelining

Pipelining in F#

```
[| 83uy; 80uy; 79uy; 67uy; 75uy |]
|> Encoding.ASCII.GetString(textInBytes)
|> Console.WriteLine(contents)
```

Method chaining in C#

```
var sb = new StringBuilder();
sb.Append("Hello")
  .Append(",")
  .Append("World")
  .AppendLine("!");
```

Коды ошибок


Менеджер Времени
60
1200
E 10
Темп.
Отжим
Легкая глажка
Доп. полоскание
Отложенный старт
Старт/Пауза
EWS 12612 W
Хлопок
Обычная стирка
Экон.стирка
Синтетика

Демо

4 вида методов

Commands:

```
   void EnrollStudent(Student student);//not expected to fail

Result EnrollStudent(Student student);//expected to fail
```

Queries:

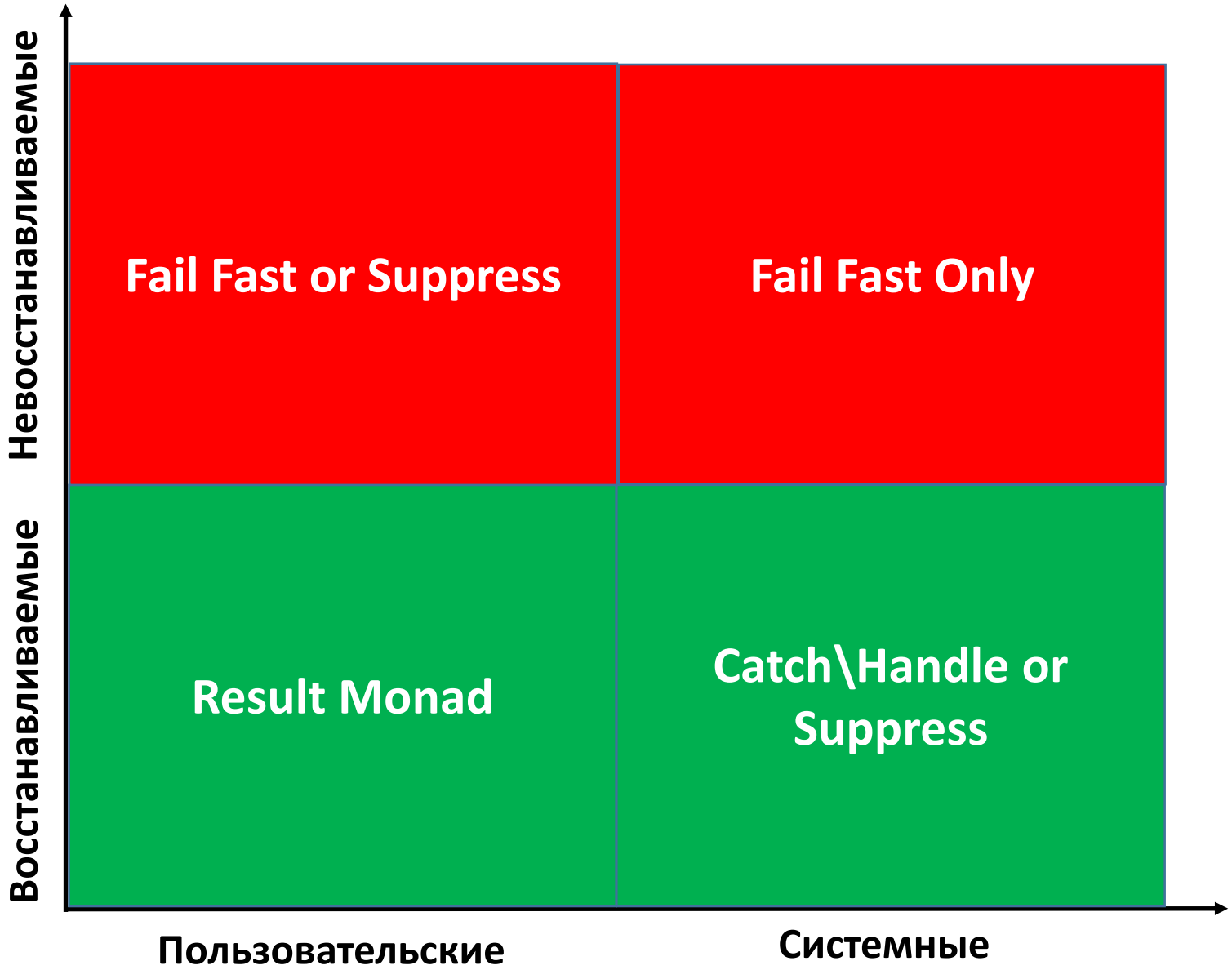
```
        Student GetStudent(string name); //not expected to fail

Result<Student> GetStudent(string name);//expected to fail
```

Итоги

Что делать?

- Отслеживайте control flow.
  Если вызов уходит в unmanaged будьте готовы к неприятностям.
- Есть три основных подхода к обработке исключений.
  Выбирайте тот, который оптимальным образом удовлетворяет требования по устойчивости конкретного приложения.

Категории ошибок – что делать?



Ссылки

Владимир Хориков

http://pluralsight.com/courses/csharp-applying-functional-principles
«Applying Functional Principles in C#» - видео курс

https://github.com/vkhorikov/CSharpFunctionalExtensions
Класс Result и полезные расширения – исходники

https://www.nuget.org/packages/CSharpFunctionalExtensions/
Класс Result и полезные расширения – NuGet package

https://www.visualstudio.com/en-us/products/visual-studio-dev-essentials-vs.aspx

Ссылки

https://ericlippert.com/2014/03/03/living-with-unchecked-exceptions/
Статья Эрика Липперта в двух частях про исключения

https://vimeo.com/97344498
Scott Wlaschin - Railway Oriented Programming — error handling in functional languages

http://joeduffyblog.com/2016/02/07/the-error-model/
Очень крутая статья Joe Duffy об обработке ошибок

http://www.artima.com/intv/handcuffs.html
Андерс Хейлсберг об исключениях

https://www.nuget.org/packages/EnterpriseLibrary.ExceptionHandling/
Enterprise Exception Handling Block

https://github.com/App-vNext/Polly
Polly

http://sergeyteplyakov.blogspot.ru/2016/06/semantic-of-exception.html
Статья Теплякова об исключениях

Спасибо за внимание!

Илья Фофанов

ENGINEER SPOCK

http://engineerspock.com

https://www.udemy.com/user/eliasfofanov/

https://habrahabr.ru/users/engineerspock/