

Параноидальное Программирование

Март 31, 2018

SAR
DOT
NET

О себе



ILYA SHETININ

Lead Software Engineer

Back-end .NET Developer

5 лет в EPAM

Специализация:

.NET, Desktop development

Бизнес-область:

Страхование

План

- Контракты.
- Параноидальное программирование.
- Null - хорошо или плохо?
- Должно ли приложение падать?
- Кто ответственен за код?



Данные

Контракты



Контракты

```
/// <summary>  
///  Переворачивает строку.  
/// </summary>  
string ReverseString(string stringToReverse)
```

Контракты

- Null
- String.Empty
- Перевернутую строку
- ArgumentException

Контракты

```
/// <summary>  
/// Ищет человека с соответствующим ID в базе данных.  
/// </summary>  
Person FindPerson(int personID)
```

Контракты

- Null
- Person
- InvalidPersonIdException
- PersonNotFoundException
- DatabaseNotAvailableException

```
bool DoesFtpDirectoryExist(string url,  
    string directory,  
    NetworkCredential credentials)
```

Контракты

- True/False
- `UrlNotAvailableException`
- `NotAuthorizedException`

Контракты

- True
- UrlNotAvailableException
- NotAuthorizedException
- **DirectoryNotExistsException**

Контракты

```
/// <summary>  
/// Валидирует заказ  
/// </summary>  
bool? Validate(Order order);
```

Контракты



Параноидальное программирование

Параноидальное программирование

```
public string GetCustomerName(Customer customer)
{
    return customer.Name;
}
```

Параноидальное программирование

```
public string GetCustomerName(Customer customer)
{
    return customer != null ? customer.Name : null;
}
```

Параноидальное программирование

```
public string GetOrderDescription(Order order, Customer customer)
{
    return order.Date + " " + GetCustomerName(customer).Substring(0, 10) + "...";
}
```

Параноидальное программирование

```
public string GetOrderDescription(Order order, Customer customer)
{
    string customerName = null;
    if (customer != null)
    {
        customerName = GetCustomerName(customer).Substring(0, 10);
    }

    return order.Date + " " + customerName + "...";
}
```

Параноидальное программирование

```
public string GetOrderDescription(Order order, Customer customer)
{
    string customerName = null;
    if (customer != null)
    {
        customerName = GetCustomerName(customer);
        if (!String.IsNullOrEmpty(customerName))
        {
            customerName = customerName.Substring(0, 10);
        }
    }

    return order.Date + " " + customerName + "...";
}
```

Параноидальное программирование

```
public string GetOrderDescription(Order order, Customer customer)
{
    string customerName = null;
    if (customer != null)
    {
        customerName = GetCustomerName(customer);
        if (!String.IsNullOrEmpty(customerName))
        {
            customerName = customerName.Substring(0, 10);
        }
    }

    string orderDate = order != null;
    if (order != null)
    {
        orderDate = order.Date.ToString();
    }

    return orderDate + " " + customerName + "...";
}
```

Параноидальное программирование

```
public bool Validate(Order order)
{
    if (order == null)
        return true;

    // Здесь происходит валидация
}
```

Параноидальное программирование

```
public bool Validate(Order order)
{
    if (order == null)
        return true;

    if (order.Customer == null)
        return false;

    // Здесь происходит валидация
}
```


Параноидальное программирование

```
public bool Validate(Order order)
{
    if (order == null) return true;
    if (order.Customer == null) return false;

    try
    {
        // Здесь происходит валидация
    }
    catch (Exception ex)
    {
        return false;
    }
}
```

Параноидальное программирование

```
public void ProcessOrder(Order order)
{
    var resultA = DoA(order);
    var resultB = DoB(order, resultA);

    SaveToDb(order, resultA, resultB);
}
```

Проблемы параноидального программирования

- Null-reference exception
- Проверки на null заразительны
- Ошибки скрываются
- Плохие данные могут попасть в бд
- Неоднозначные контракты

Параноидальное программирование

```
public bool Validate(Order order)
{
    if (order == null) return true;
    if (order.Customer == null) return false;

    try
    {
        // Здесь происходит валидация
    }
    catch (Exception ex)
    {
        return false;
    }
}
```

Параноидальное программирование

```
public bool Validate(Order order)
{
    if (order == null) return true;
    if (order.Customer == null) return false;

    try
    {
        // Здесь происходит валидация
    }
    catch (ArgumentException ex)
    {
        return false;
    }
}
```

Параноидальное программирование

```
public bool Validate(Order order)
{
    if (order == null) return true;
    if (order.Customer == null) return false;

    try
    {
        // Здесь происходит валидация
    }
    catch (Exception ex)
    {
        MyFavoriteLogger.Log(ex);
        throw;
    }
}
```

Исключительные vs неисключительные ситуации



Параноидальное программирование

```
public bool Validate(Order order)
{
    if (order == null)
        return true;

    // Представим, что здесь происходит валидация
}
```



```
public bool Validate(Order order)
{
    if (order == null)
        throw new ArgumentNullException(nameof(order));

    // Представим, что здесь происходит валидация
}
```

Пользователь увидел ошибку



FAIL FAST

Защитное программирование

Защитное программирование основано на важной предпосылке: худшее, что может сделать модуль, это принять неправильные входные данные и затем вернуть неверный, но правдоподобный результат

Когда приложение должно упасть



Fail Fast после отображения ошибки

- Сбрасываем состояние.
- Завершаем работу приложения.

Плюсы Fail Fast

- Быстрее узнаем об ошибках.
- Легко понять код.
- Качество кода лучше.
- В БД корректные данные.
- Мы уверены, что ПО работает правильно.

Корректность vs Отказоустойчивость



Ссылки

- Параноидальное программирование
<http://johannesbrodwall.com/2013/09/25/offensive-programming/>
- Fail Fast
<http://enterprisecraftsmanship.com/2015/09/15/fail-fast-principle/>
- Корректность и отказоустойчивость
<https://rdingwall.com/2010/02/10/correctness-vs-robustness/>
- Защитное программирование
<http://blog.ploeh.dk/2013/07/08/defensive-coding/>
<https://dzone.com/articles/defensive-programming-just>

**СПАСИБО ЗА
ВНИМАНИЕ!**

CONTACT ME



Ilya_Shetinin@epam.com



Ilya_shetinin