



# Advanced C#

LUKÁŠ VAVREK

# Outline

Generic Programming

LINQ

# Generic Programming



# Awesome storage class

```
class IntStorage
{
    private int[] _data;

    public void add(int n) { ... }
    public int get(int index) { ... }
}
```

# We should support strings

```
class StringStorage
{
    private string[] _data;

    public void add(string n) { ... }
    public string get(int index) { ... }
}
```

# What is the problem?



- ⟩ Repetitive code
- ⟩ Shared bugs vs Specific bugs
- ⟩ Classes have nothing in common

# Generic code

```
class Storage<T>
{
    private T[] _data;

    public void add(T n) { ... }
    public T get(int index) { ... }
}
```

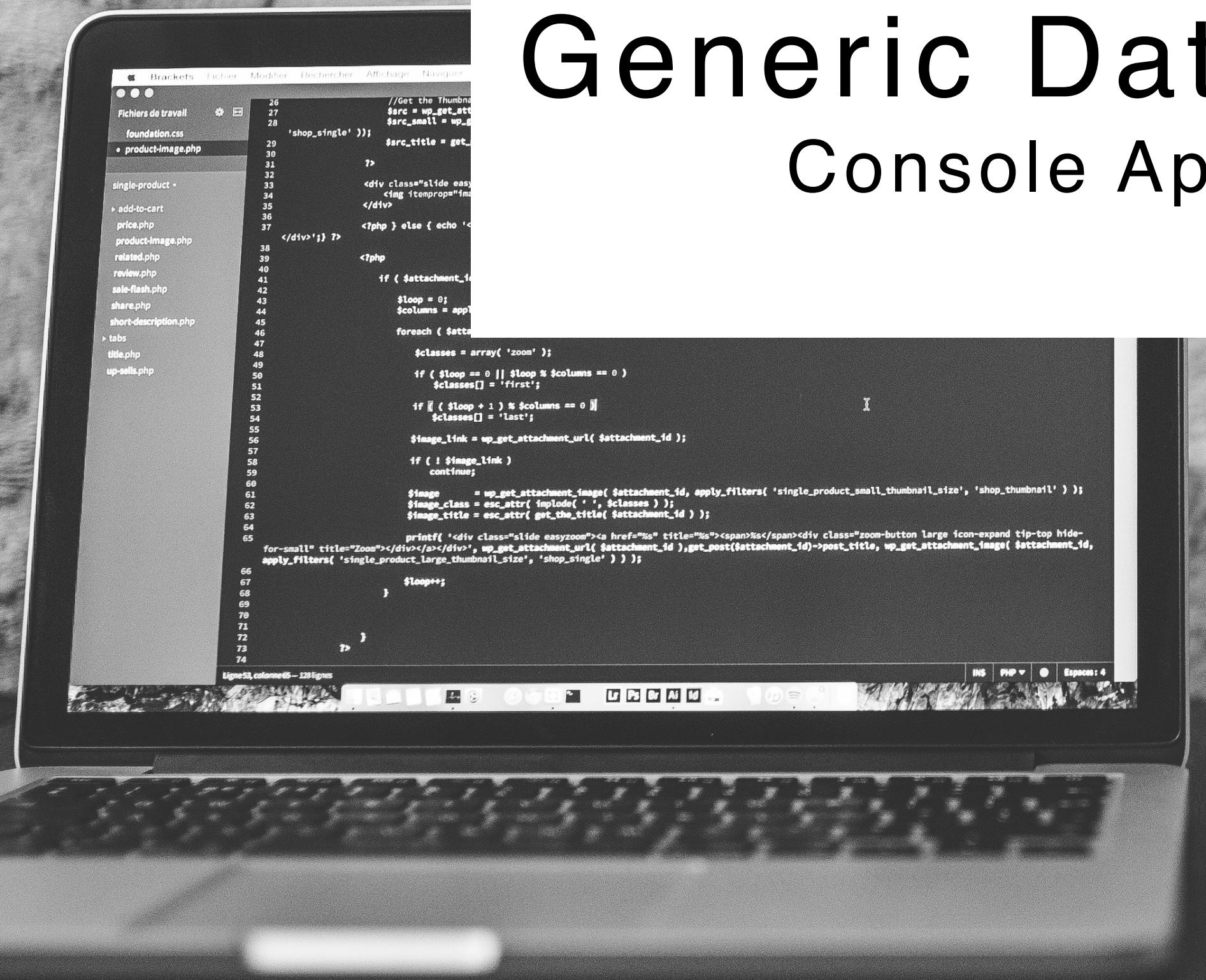
# Instantiation

```
var intStorage = new Storage<int>()  
var stringStorage = new Storage<string>()
```

# DEMO

# Generic Data Storage

## Console Application



# LINQ



The image shows a person working on a laptop, with a smartphone connected via a USB cable. The laptop screen displays code in an Android Studio interface, specifically Java code for a service named `LiveEventsService`. The code includes methods for starting and stopping polling for updates, and managing pending intents and alarm managers. The background is a blurred office environment.

```
public void startPollingForUpdates(@NotNull String uid) {
    Timber.d("startPollingForUpdates(), with this uid: %s", uid);
    long interval = 1000 * 60; // check every minute
    long start = System.currentTimeMillis() + interval;
    Intent i = LiveEventsService.getCheckLiveEventIntent(this, uid);
    pi = PendingIntent.getService(this, 0, i, 0);
    AlarmManager alarmManager = (AlarmManager) getSystemService(Context.ALARM_SERVICE);
    alarmManager.setRepeating(AlarmManager.RTC_WAKEUP, start, interval, pi);
}

public void stopPollingForUpdates() {
    Timber.d("stopPollingForUpdates()");
    if (pi != null) {
        AlarmManager alarmManager = (AlarmManager) getSystemService(Context.ALARM_SERVICE);
```

# SQL

SQL (Structured Query Language) is a domain-specific language used in programming and designed for managing data held in a relational database management systems (RDBMS), or for stream processing in a relational data stream management system (RDSMS).

# SQL

```
SELECT p.Name AS ProductName,  
NonDiscountSales = (OrderQty * UnitPrice),  
Discounts = ((OrderQty * UnitPrice) *  
UnitPriceDiscount)  
FROM Production.Product AS p  
INNER JOIN Sales.SalesOrderDetail AS sod  
ON p.ProductID = sod.ProductID  
ORDER BY ProductName DESC;
```

# LINQ

LINQ (Language Integrated Query) is a Microsoft .NET Framework component that adds native data querying capabilities to .NET languages. LINQ extends the language by the addition of query expressions, which are akin to SQL statements and can be used to conveniently extract and process data from arrays, enumerable classes, XML documents, relational databases, and third-party data sources.

# LINQ

```
var query_join4 =  
    from a in svcContext.AccountSet  
    join c in svcContext.ContactSet  
        on a.PrimaryContactId.Id equals c.ContactId  
    join l in svcContext.LeadSet  
        on a.OriginatingLeadId.Id equals l.LeadId  
    select new  
{  
    contact_name = c.FullName,  
    account_name = a.Name,  
    lead_name = l.FullName  
};
```

# LINQ

```
var studentNames = studentList.Where(s => s.Age > 18)
    .Select(s => s)
    .Where(st => st.StandardID > 0)
    .Select(s => s.StudentName);
```

# DEMO

## Querying Data Console Application



```
function generate_product_thumbnails() {
    // Get the Thumbnail
    $src = wp_get_attachment_image_src($attachment_id, 'shop_thumbnail');
    $src_mall = wp_get_attachment_image_src($attachment_id, 'shop_single');

    // Output the thumbnail
    echo '';

    // Output the large thumbnail
    echo '';

    // Output the title
    echo '

### ' . esc_attr($product->post_title) . '

';

    // Output the price
    echo '

' . esc_html($product->post_content) . '

';
}
```



# Q & A

