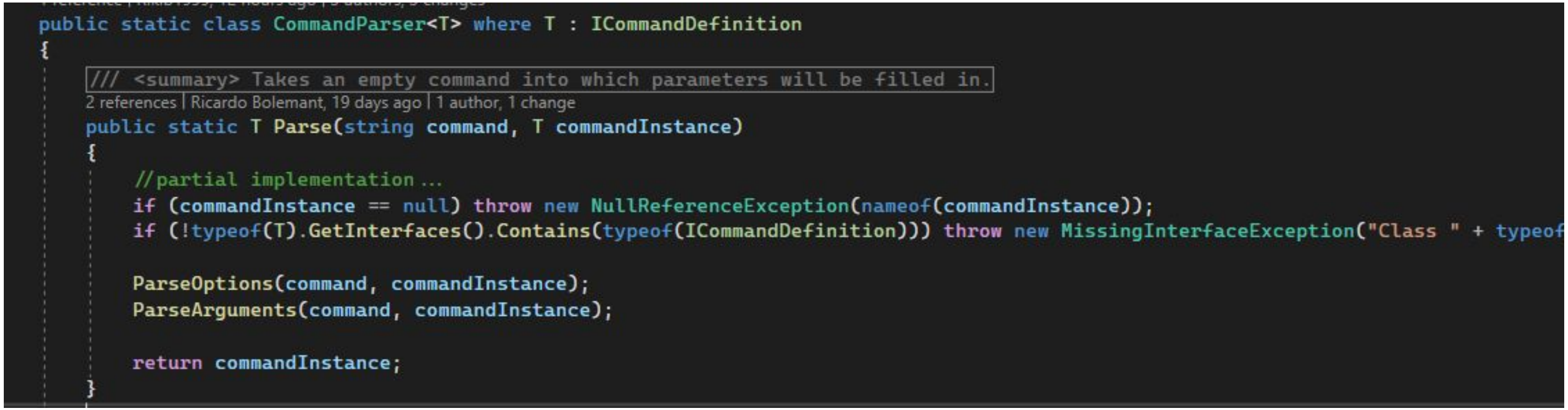
Command parser API

Boris Kapustík, Ricardo Bolemant

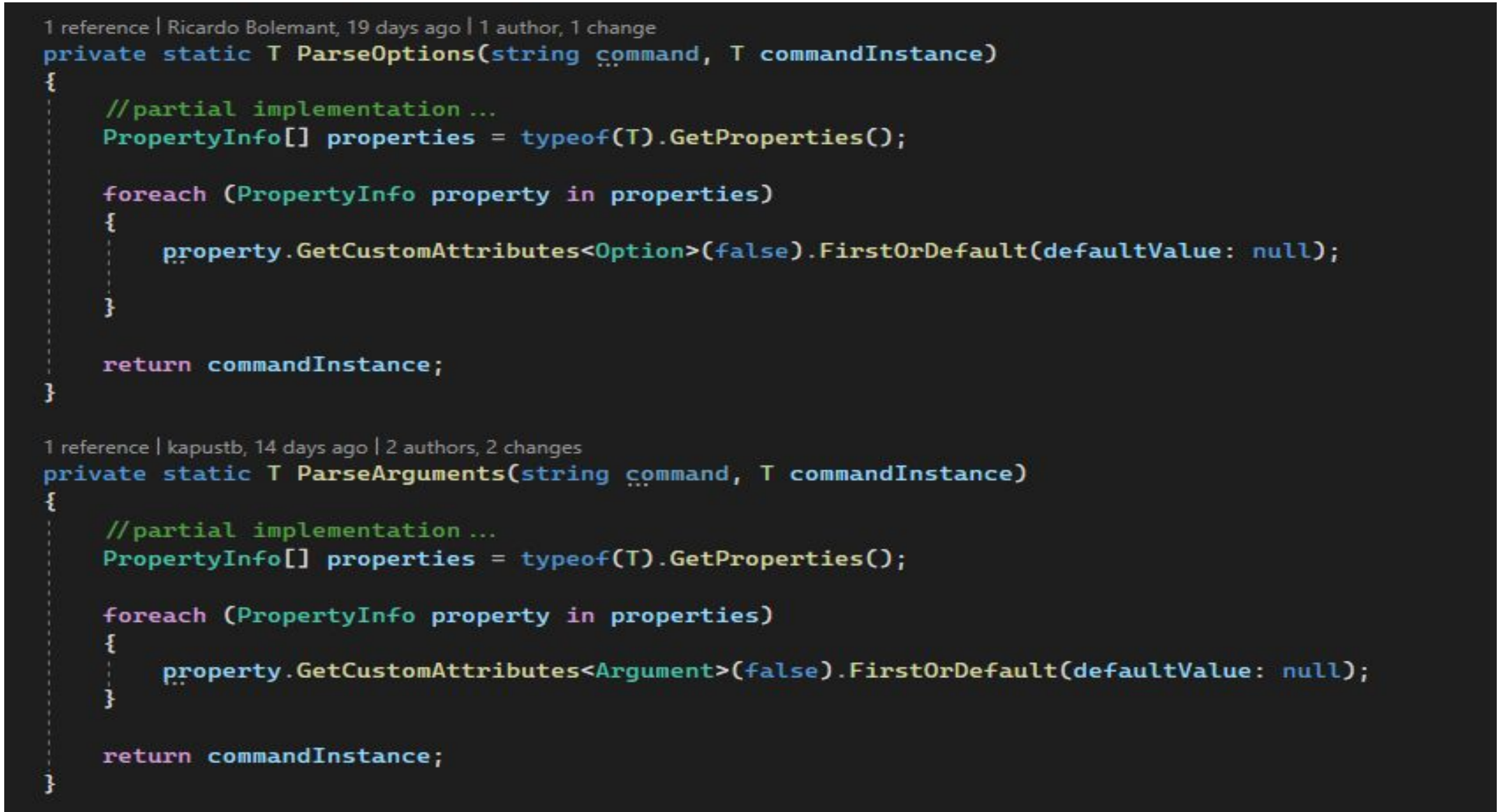
Generický CommandParser



```
public static class CommandParser<T> where T : ICommandDefinition
{
    /// <summary> Takes an empty command into which parameters will be filled in.
    2 references | Ricardo Bolemant, 19 days ago | 1 author, 1 change
    public static T Parse(string command, T commandInstance)
    {
        //partial implementation ...
        if (commandInstance == null) throw new NullReferenceException(nameof(commandInstance));
        if (!typeof(T).GetInterfaces().Contains(typeof(ICommandDefinition))) throw new MissingInterfaceException("Class " + typeof

        ParseOptions(command, commandInstance);
        ParseArguments(command, commandInstance);

        return commandInstance;
    }
```

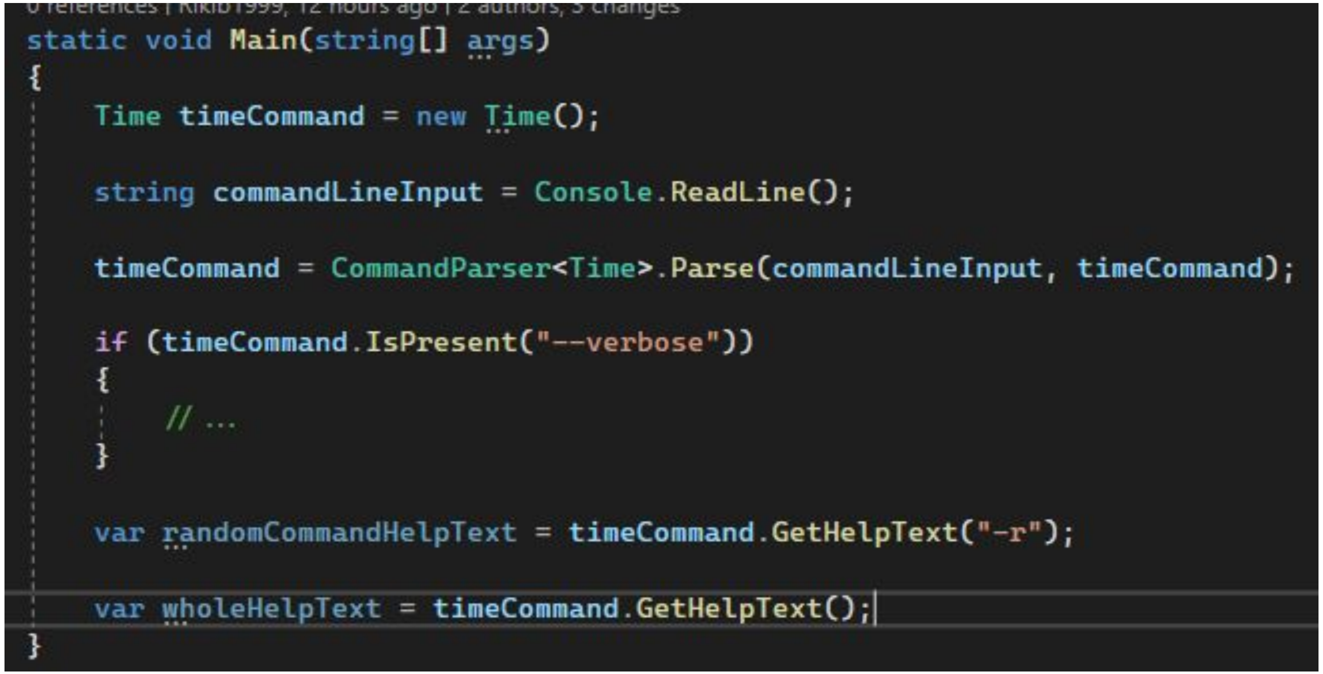
- Užívateľovi sa vráti objekt s naparsovanými a validovanými hodnotami

Implementácia pomocou reflection



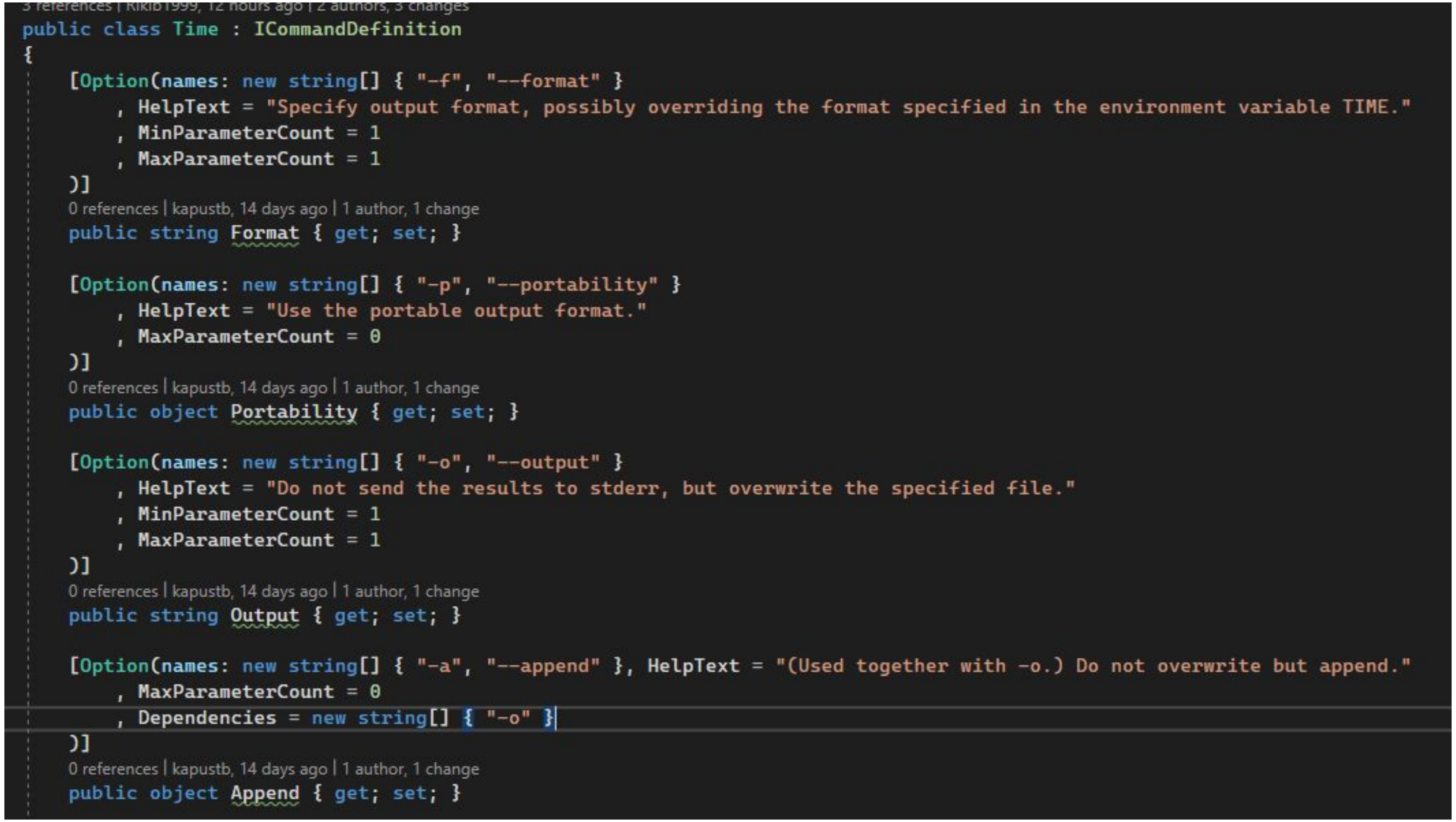
```
1 reference | Ricardo Bolemant, 19 days ago | 1 author, 1 change
private static T ParseOptions(string command, T commandInstance)
{
    //partial implementation ...
    PropertyInfo[] properties = typeof(T).GetProperties();

    foreach (PropertyInfo property in properties)
    {
        property.GetCustomAttributes<Option>(false).FirstOrDefault(defaultValue: null);

    }

    return commandInstance;
}

1 reference | kapustb, 14 days ago | 2 authors, 2 changes
private static T ParseArguments(string command, T commandInstance)
{
    //partial implementation ...
    PropertyInfo[] properties = typeof(T).GetProperties();

    foreach (PropertyInfo property in properties)
    {
        property.GetCustomAttributes<Argument>(false).FirstOrDefault(defaultValue: null);
    }

    return commandInstance;
}
```

Použitie API



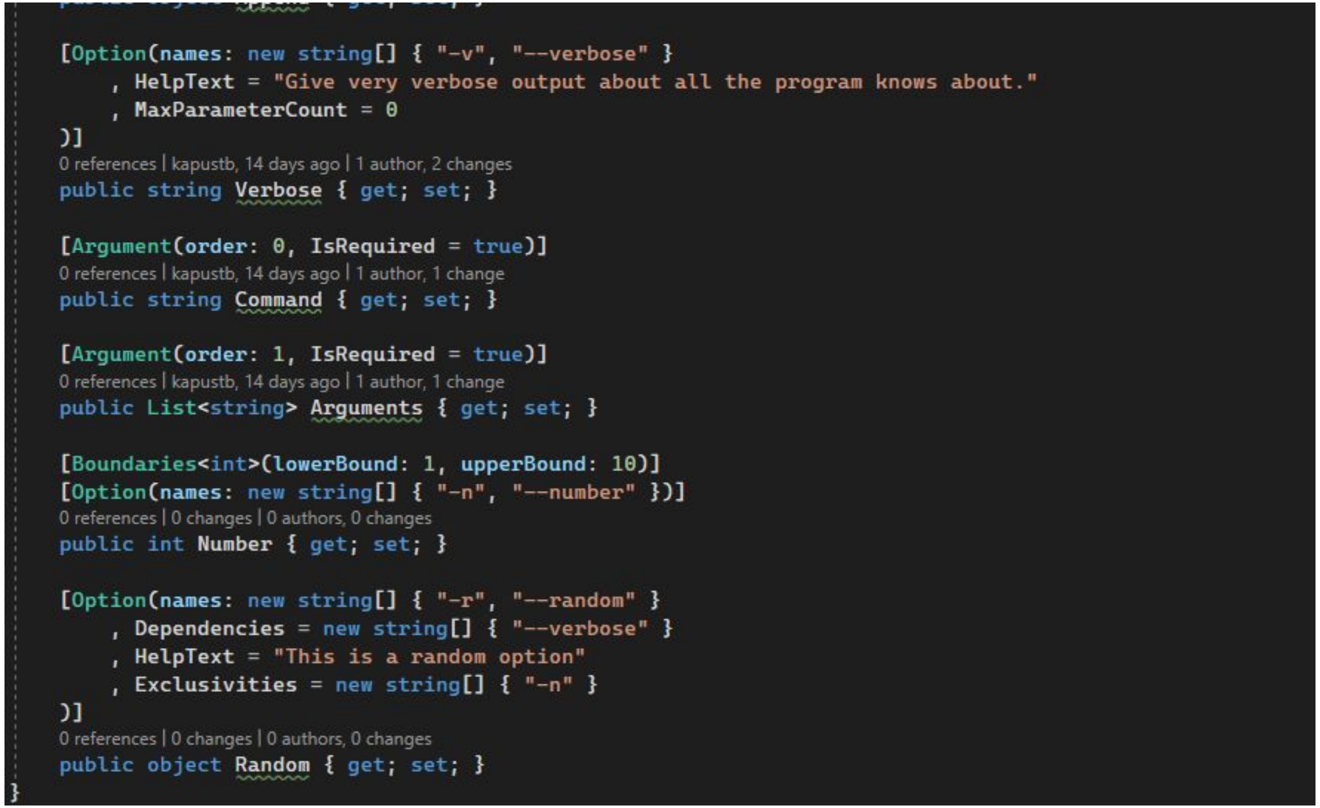
```
0 references | KikiD1999, 12 hours ago | 2 authors, 3 changes
static void Main(string[] args)
{
    Time timeCommand = new Time();

    string commandLineInput = Console.ReadLine();

    timeCommand = CommandParser<Time>.Parse(commandLineInput, timeCommand);

    if (timeCommand.IsPresent("--verbose"))
    {
        // ...
    }

    var randomCommandHelpText = timeCommand.GetHelpText("-r");

    var wholeHelpText = timeCommand.GetHelpText();
}
```

Použitie API



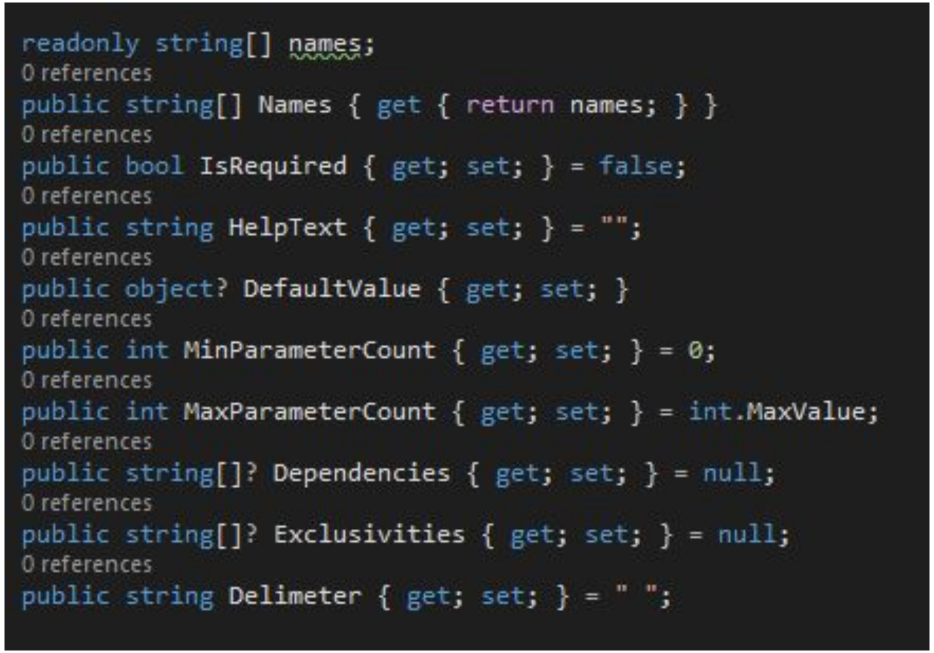
```
3 references | Rikib1999, 12 hours ago | 2 authors, 3 changes
public class Time : ICommandDefinition
{
    [Option(names: new string[] { "-f", "--format" }
        , HelpText = "Specify output format, possibly overriding the format specified in the environment variable TIME."
        , MinParameterCount = 1
        , MaxParameterCount = 1
    )]
    0 references | kapustb, 14 days ago | 1 author, 1 change
    public string Format { get; set; }

    [Option(names: new string[] { "-p", "--portability" }
        , HelpText = "Use the portable output format."
        , MaxParameterCount = 0
    )]
    0 references | kapustb, 14 days ago | 1 author, 1 change
    public object Portability { get; set; }

    [Option(names: new string[] { "-o", "--output" }
        , HelpText = "Do not send the results to stderr, but overwrite the specified file."
        , MinParameterCount = 1
        , MaxParameterCount = 1
    )]
    0 references | kapustb, 14 days ago | 1 author, 1 change
    public string Output { get; set; }

    [Option(names: new string[] { "-a", "--append" }, HelpText = "(Used together with -o.) Do not overwrite but append."
        , MaxParameterCount = 0
        , Dependencies = new string[] { "-o" }
    )]
    0 references | kapustb, 14 days ago | 1 author, 1 change
    public object Append { get; set; }
```

Použitie API



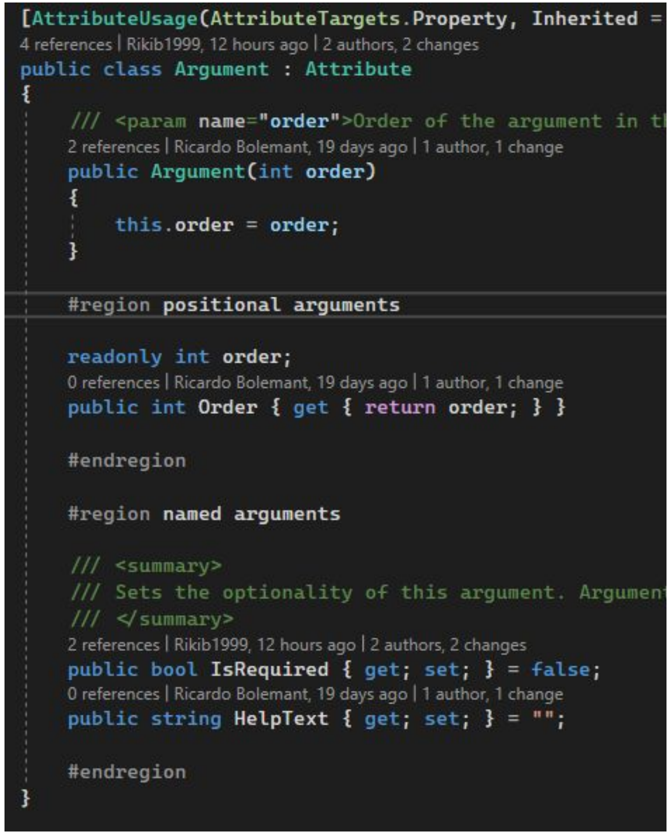
```
    [Option(names: new string[] { "-v", "--verbose" }
        , HelpText = "Give very verbose output about all the program knows about."
        , MaxParameterCount = 0
    )]
    0 references | kapustb, 14 days ago | 1 author, 2 changes
    public string Verbose { get; set; }

    [Argument(order: 0, IsRequired = true)]
    0 references | kapustb, 14 days ago | 1 author, 1 change
    public string Command { get; set; }

    [Argument(order: 1, IsRequired = true)]
    0 references | kapustb, 14 days ago | 1 author, 1 change
    public List<string> Arguments { get; set; }

    [Boundaries<int>(lowerBound: 1, upperBound: 10)]
    [Option(names: new string[] { "-n", "--number" })]
    0 references | 0 changes | 0 authors, 0 changes
    public int Number { get; set; }

    [Option(names: new string[] { "-r", "--random" }
        , Dependencies = new string[] { "--verbose" }
        , HelpText = "This is a random option"
        , Exclusivities = new string[] { "-n" }
    )]
    0 references | 0 changes | 0 authors, 0 changes
    public object Random { get; set; }
}
```

Option



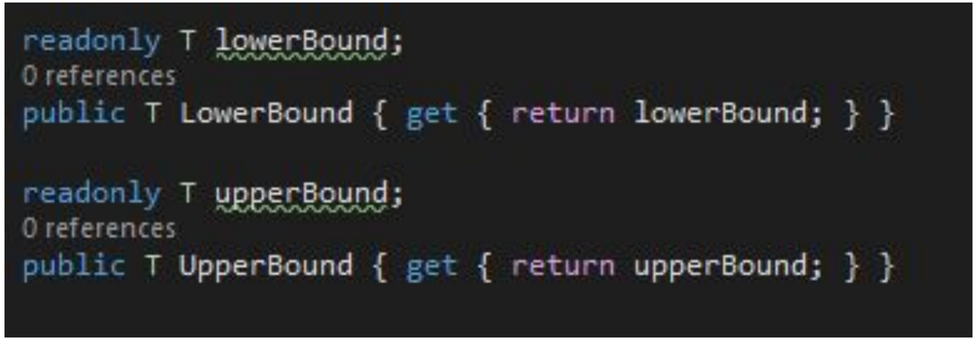
```
readonly string[] names;
0 references
public string[] Names { get { return names; } }
0 references
public bool IsRequired { get; set; } = false;
0 references
public string HelpText { get; set; } = "";
0 references
public object? DefaultValue { get; set; }
0 references
public int MinParameterCount { get; set; } = 0;
0 references
public int MaxParameterCount { get; set; } = int.MaxValue;
0 references
public string[]? Dependencies { get; set; } = null;
0 references
public string[]? Exclusivities { get; set; } = null;
0 references
public string Delimeter { get; set; } = " ";
```

Argument



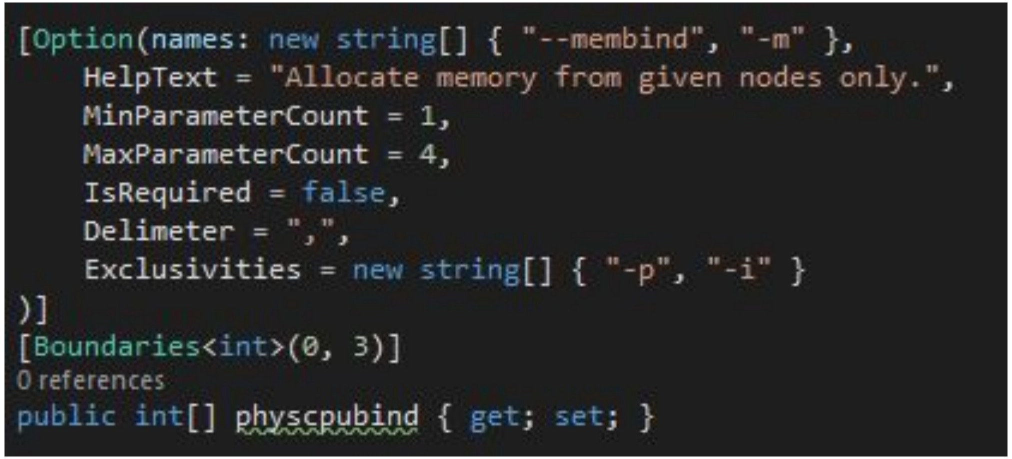
```
[AttributeUsage(AttributeTargets.Property, Inherited =
4 references | Rikib1999, 12 hours ago | 2 authors, 2 changes
public class Argument : Attribute
{
    /// <param name="order">Order of the argument in th
    2 references | Ricardo Bolemant, 19 days ago | 1 author, 1 change
    public Argument(int order)
    {
        this.order = order;
    }

    #region positional arguments

    readonly int order;
    0 references | Ricardo Bolemant, 19 days ago | 1 author, 1 change
    public int Order { get { return order; } }

    #endregion

    #region named arguments

    /// <summary>
    /// Sets the optionality of this argument. Argument
    /// </summary>
    2 references | Rikib1999, 12 hours ago | 2 authors, 2 changes
    public bool IsRequired { get; set; } = false;
    0 references | Ricardo Bolemant, 19 days ago | 1 author, 1 change
    public string HelpText { get; set; } = "";

    #endregion
}
```

Boundaries

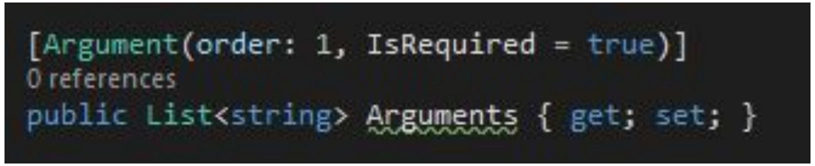


```
readonly T lowerBound;
0 references
public T LowerBound { get { return lowerBound; } }

readonly T upperBound;
0 references
public T UpperBound { get { return upperBound; } }
```

Attribute



```
[Option(names: new string[] { "--membind", "-m" },
    HelpText = "Allocate memory from given nodes only.",
    MinParameterCount = 1,
    MaxParameterCount = 4,
    IsRequired = false,
    Delimeter = ",",
    Exclusivities = new string[] { "-p", "-i" }
)]
[Boundaries<int>(0, 3)]
0 references
public int[] physcpubind { get; set; }
```



```
[Argument(order: 1, IsRequired = true)]
0 references
public List<string> Arguments { get; set; }
```