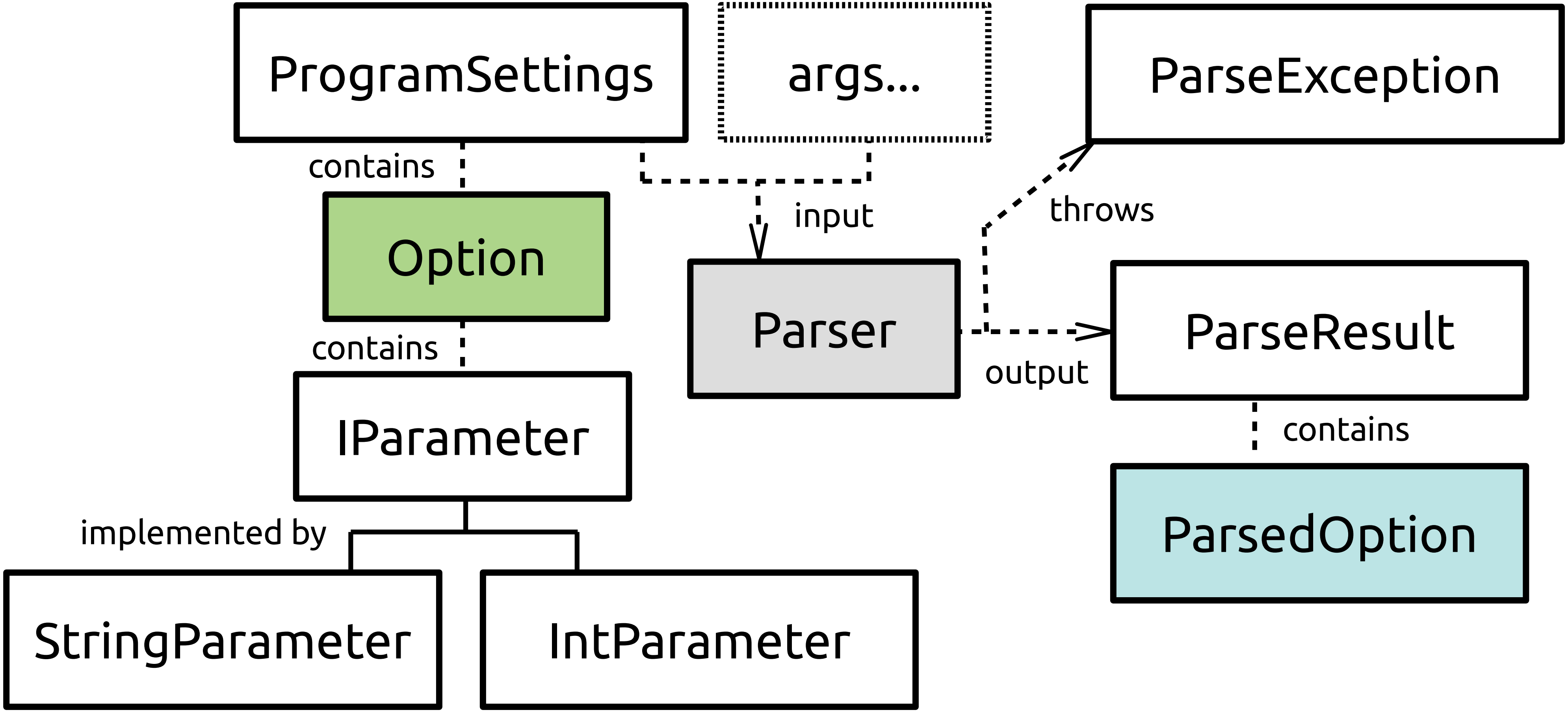
CommandLineParser

API presentation

25 Mar 2019


ProgramSettings
args...
ParseException
contains
input
throws
Option
Parser
ParseResult
contains
output
contains
IParameter
ParsedOption
implemented by
StringParameter
IntParameter

```csharp
// create ProgramSettings
ProgramSettings s = new ProgramSettings("time");

// add options to settings, with or without parameters
s.AddOption('f', "format", isMandatory: false, parameter:
new StringParameter(isMandatory: true),
    "Specify output format (...)");
s.AddOption('p', "portability", isMandatory: false, parameter: null,
    "Use the portable output format.");
s.AddOption('o', "output", isMandatory: false, parameter:
new StringParameter(isMandatory: true),
    "Do not send the results to stderr (...)");
s.AddOption('a', "append", isMandatory: false, parameter: null,
    "(Used together with -o.) Do not overwrite but append.");
s.AddOption('v', "verbose", isMandatory: false, parameter: null,
    "Give very verbose output about all the program knows about.");
s.AddOption('V', "version", isMandatory: false, parameter: null,
    "Print version information on standard output (...)");
s.AddOption(shortName: null, "help", isMandatory: false, parameter: null,
    "Print a usage message on standard output  (...)");
```

```
// both of these options must be present if first option is present
s.AddDependency("a", "o");

// example: conflicting options ~ only one of the following may be present
// s.AddConflictingOptions("a", "version", "o");

// parse using defined settings
ParseResult result = Parser.Parse(args, s);

// get parameter value for specific option (explicit cast needed)
string output = (string)result.GetParameterValue("o");

// find out whether this option was present; good for non-parametric options
result.WasParsed("v");
```

```
// enumerate all parsed options
foreach (var item in result.ParsedOptions)
{
    if (item.Names.Contains("V"))
    {
        // printVersionInfo();
    }
    else if (item.Names.Contains("o"))
    {
        string outFile = (string)item.Value;
        // printOutput(outFile);
    }
    // ...
}

// get all the plain arguments in the same order as on the command line
List<string> plainArgs = result.PlainArguments;

// print help using specified TextWriter
settings.PrintHelp(Console.Out);
settings.PrintHelp(new System.IO.StreamWriter("myfile.txt"));
```

```
public interface IParameter
{
    bool IsMandatory { get; }
    bool TryParse(string value, out object result);
}
```

□

```
public class IntParameter : IParameter
{
    public bool IsMandatory { get; }

    public IntParameter(bool isMandatory,
        int lowerBound = int.MinValue, int upperBound = int.MaxValue) { … }

    public bool TryParse(string value, out object result) { … }
}
```

□