# Parsing data ,Levels of Effort, Tools for Gathering Clues

**The structure of a parser**

Generally speaking, parsing, or syntax analysis is the process of analyzing a string of symbols in a language, conforming to the rules of formal grammar. Parsing in terms of data analysis extends this definition into a two-step process, in which the parser is programmatically instructed on which data to read, analyze or transform. The result usually is a more structured format.

The two components that make up a data parser are known as **lexical analysis** and **syntactic analysis**. Some parsers also offer a semantic analysis component, which takes the remaining parsed and structured data and applies meaning. For instance, a semantic analysis may filter the data further such as: positive or negative, complete or incomplete, etc. Semantic analysis may further enhance the data analysis process, but this is not always the case.

It's important to note that while some parsers yield helpful insights, semantic analysis is not inherently built into the structure of most parsers, due to the favored practice of human semantic analysis. This step should be considered an additional step, should you choose it, as it may complement your business goals.

Data parsing has two main steps. Together these steps transform a string of unstructured data into a tree of data, whose rules and syntax are built into the tree's structure. Let's explore both steps.

**Lexical analysis**

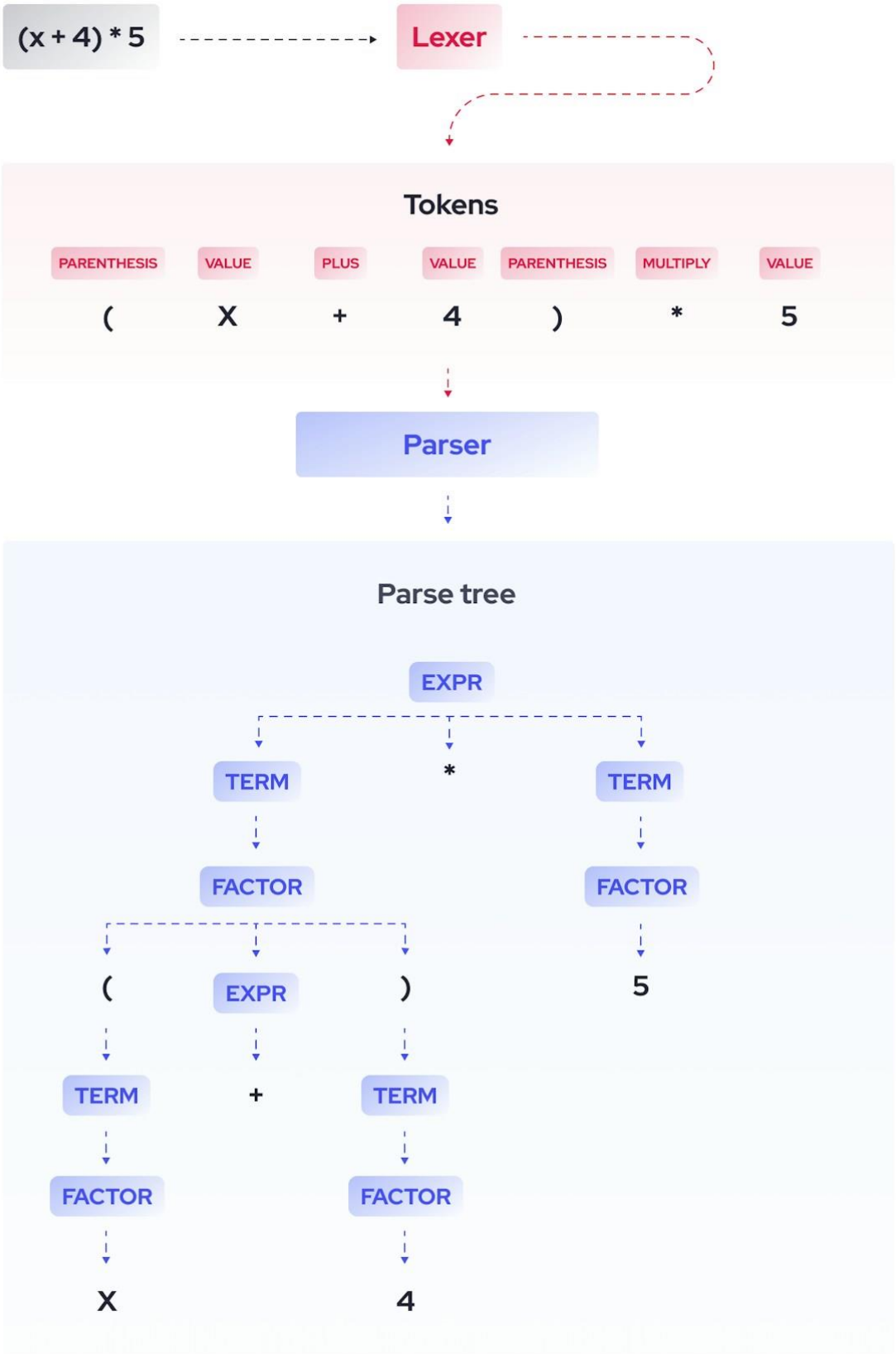
The first step of data parsing occurs during lexical analysis. Lexical analysis, in its simplest form, creates **tokens**, with a **lexer**, from a sequence of characters that enter the parser as a string of raw unstructured data. Oftentimes this string of data enters the parser in an HTML format. The parser creates tokens by utilizing lexical units (keywords, identifiers, and delimiters) while simultaneously ignoring lexically irrelevant information (such as white spaces and comments).

The parser then discards any irrelevant tokens, established during the lexical process. The remainder of the parsing process falls into the category of syntactic analysis.

**Syntactic analysis**

The syntactic analysis component of data parsing consists of **parse tree** building. What does this mean? A **parser** takes the aforementioned tokens and arranges them into a parse tree, in which any irrelevant tokens are captured in the nesting structure of the tree itself. Irrelevant tokens include elements such as parenthesis, curly braces, and semicolons.

To further illustrate this concept, let's look at a common mathematical example such as (x + 4) * 5. The diagram below illustrates the general process of data parsing in the form of a lexer and a parser.


(x + 4) * 5
Lexer
Tokens
PARENTHESIS
VALUE
PLUS
VALUE
PARENTHESIS
MULTIPLY
VALUE
( X + 4 ) * 5
Parser
Parse tree
EXPR
TERM
*
TERM
FACTOR
FACTOR
( EXPR ) 5
TERM
+
TERM
FACTOR
FACTOR
X
4

In the case of understanding this illustration in terms of real-life applications, a data parser would apply this same logic to more complex data. In its simplest form, a data parser would create tokens from data in an HTML document and transform said tokens into a parse tree.

**Parsing technologies**

So what technologies and languages can parsing methods be used with? Due to the extremely flexible nature of data parsers, they can be used in conjunction with many technologies. Some of these technologies include:

Scripting languages

- Scripting languages create a series of commands that can be executed without the need for compiling.
- These languages are seen in web applications, games, and multimedia, as well as plugins and extensions.

HTML and XML

- Also known as Hypertext Markup Language, HTML is used to create web pages and web page applications that display data.
- Similarly, XML (eXtensible Markup Language) is used for the transportation of data within web pages and web applications.

Interactive data language

- These languages are used for interactive processing of large amounts of data, including interactive processing.
- Its application is accepted widely in space sciences and solar physics.

Modeling languages

- Are used to specify system requirements, structures, and behaviors.
- These languages are used by interested parties (developers, analysts, investors) to understand the workings of the system being modeled.

SQL and database languages

- Structured Query Language, or SQL, is a programming language used for managing data in database systems.

HTTPS and Internet Protocols

- Hypertext Transfer Protocol and other Internet Protocol languages are used as a communication protocol and are the foundation of data communication for the world wide web.

**Data parsing use cases**

Web scraping

Before beginning the data parsing process, companies must acquire substantial-quality data. Data extraction in the form of **web scraping** is an essential prerequisite to the data scraping process. During the web scraping process, a scraper retrieves an unparsed HTML document (see above for uses) with extraneous information, such as list tags. In this case, a parser would transform the data taken from a web scraper and remove such tags, in addition to performing other basic tasks previously mentioned (token creation and syntactic analysis).

Workflow optimization

**Workflow optimization** is built into the basic function of data parsers. By transforming unstructured data into data that is more readable, companies can improve their workflow.

Some teams that may see a substantial increase in productivity include data analysts, programmers, marketers, and investors.

Investment analysis

Acquiring data for investment efforts such as equity research, evaluating start-ups, earnings forecasts, and **competitive analysis** is a time-consuming effort. Likewise, data analysis requires access to substantial data processing resources. One may shortcut these resources by utilizing web scraping tools in conjunction with a data parser. This will optimize workflow, and will consequently allow you to direct resources elsewhere or focus them on a more in-depth data analysis.

More specifically, investors and data analysts can harness data parsing in a way that provides better insights for business decisions. Investors, hedge funds, and other professionals that evaluate start-ups, predict earnings, and even monitor social sentiment rely on web-scraping followed by robust data parsing for up-to-date market insights.

**In-house or outsource?**

There are many reasons to build or buy a data parser. Let’s take a closer look at each option.

In-house pros

There are many benefits to building your own parser. Firstly, building your own parser lends you more control over the specifications of your data parser. As mentioned previously, it’s important to remember data parsers aren’t restricted to a particular data format. Instead, parsers convert one data format into another. Further, how the data is converted is dependent upon how the parser was built. For this reason, in-house parsers are beneficial due to their customizable nature.

In-house cons

Similarly, sourcing your data parser internally will allow you complete control over maintenance and updates. This method also has the potential to be more cost-efficient. However, there are a few cons to building your parser. An in-house parser will require a server strong enough for your parsing needs. Additionally, because you will have complete control over your parser, maintaining, updating, and testing your parser may consume valuable time and resources.

Outsourcing pros

Purchasing a parser can be beneficial in many ways. Primarily, because the parser would be built by a company that specializes in data extraction and parsing, you will be less likely to run into any problems. Additionally, you will have more time and resources available, as you won't need to invest in a parser team or spend any time maintaining your parser.

Outsourcing cons

Some problems that may arise with outsourcing your parser may include cost and customizability. Because data parsing providers offer a cookie-cutter solution, you will likely miss out on ways to customize your parser to your exact business needs.

**Coresignal and data parsing**

Coresignal offers robust raw data that w