Tokens and Keywords

C++ Tokens

A token is the smallest element of a C++ program that is meaningful to the compiler. The C++ parser recognizes these kinds of tokens: identifiers, keywords, literals, operators, punctuators, and other separators. A stream of these tokens makes up a translation unit.

Tokens are usually separated by "white space." White space can be one or more:

- Blanks
- Horizontal or vertical tabs
- New lines
- Formfeeds
- Comments

Syntax

token:

```
keyword
identifier
constant
operator
punctuator
```

preprocessing-token:

```
header-name
identifier
pp-number
character-constant
string-literal
operator
punctuator
```

each nonwhite-space character that cannot be one of the above

The parser separates tokens from the input stream by creating the longest token possible using the input characters in a left-to-right scan. Consider this code fragment:

```
a = i+++j;
```

The programmer who wrote the code might have intended either of these two statements:

```
a = i + (++j)
or
a = (i++) + j
```

Because the parser creates the longest token possible from the input stream, it chooses the second interpretation, making the tokens i++, +, and j.

Keywords:

Keywords implement specific C++ language features. They cannot be used as names for variables or other user-defined program elements. Most of the keywords are common to both C and C++, but some are specific to C++.

Computer programming languages, such as C++, reserve a set of words for use within the language. These words, which are called keywords, tell the compiler what your program is supposed to do. Keywords cannot be used for other purposes, such as variable names, tags, or function names.

C++ Keywords:

asm	auto	break	case	catch
char	class	const	continue	default
delete	do	double	else	enum
extern	float	for	friend	goto
if	inline	int	long	new
operator	private	protected	public	register
return	short	signed	sizeof	static
struct	switch	template	this	throw
try	typedef	union	unsigned	virtual
void	volatile	while		