1 Default minted “C” lexer

```
const double Tau = M_PI * 2;
vec3 position = vec3(0.0, 0.0, 0.0);
Atom *atom = Atom(position);
xyzzy is a generic keyword;
declaration of independence; pseudo podical;
hello_my_name_is namespace;  reserved and aloof;
```

2 Custom lexer with new keywords

```
const double Tau = M_PI * 2;
vec3 position = vec3(0.0, 0.0, 0.0);
Atom *atom = Atom(position);
xyzzy is a generic keyword;
declaration of independence; pseudo podical;
hello_my_name_is namespace;  reserved and aloof;
```

3 Latex example

```
\documentclass{article}
\usepackage[cache=false]{minted}
\usepackage[strict]{changepage}

\begin{document}
\setminted{linenos=true, frame=single}

\section{Default minted ``C'' lexer}
\inputminted{C}{example.c}

\section{Custom lexer with new keywords}
\inputminted{custom}{example.c}

\section{Latex example}
\inputminted[curlyquotes]{tex}{example.ltx}

\clearpage\setminted{linenos=false, frame=none}
\changepage{4\baselineskip}{}{-2cm}{-2cm}{}{-2cm}{}{}{}
\section{Custom keyword code}
\inputminted{python}{../pygments_custom/__init__.py}

\end{document}
```

4 Custom keyword code

```
from pygments.lexers import CLexer as mysuper          # Which lexer to inherit from.
from pygments.token import Name, Keyword

class CustomLexer(mysuper):
    """CustomLexer for pygments which extends an existing lexer with
    new keywords. They can be styled as Types, Constants, Namespaces,
    or generic Keywords. Edit the file pygments_custom/__init__.py to
    suite your taste then re-install using 'python setup.py install'.
    """

    name = 'Custom'
    aliases = ['custom']
    EXTRA = {}

    # Edit this to add new types (class, typedef) for highlighting.
    EXTRA[Keyword.Type] = ['FLAG', 'vec3', 'Atom' ]

    # The following are less commonly used and can be simply commented out.
    EXTRA[Keyword.Constant] = [ 'M_PI', 'Tau' ]
    EXTRA[Keyword.Declaration] = [ 'declaration', 'System' ]
    EXTRA[Keyword.Namespace] = [ 'hello_my_name_is' ]
    EXTRA[Keyword.Pseudo] = [ 'pseudo' ]
    EXTRA[Keyword.Removed] = [ 'removed' ]
    EXTRA[Keyword.Reserved] = [ 'reserved' ]

    # Highlight these words as generic keywords.
    EXTRA[Keyword] = [ 'xyzzy', 'plugh' ]


    def get_tokens_unprocessed(self, text, stack=('root',)):
        for index, token, value in mysuper.get_tokens_unprocessed(self, text, stack):
            if token is Name:
                for key in self.EXTRA:
                    if value in self.EXTRA[key]:
                        token=key
                        break

            yield index, token, value


if __name__ == '__main__':
    print( "testing" )
    x = CustomLexer()
    for y in x.get_tokens_unprocessed( "M_PI", "hello_my_name is", "removed" ):
        print(y)
    for y in x.get_tokens_unprocessed( "vec3 x,y,z; System;", "reserved", Type):
        print(y)
```