# PlainTEX Verbatim

Benjamin T. Shepard

July 9, 2022

An inline verbatim such as `this is \inline` can be made with the code `|...|`. If visible spaces are desired, use `@|...|` instead, which will produce `this␣is␣\inline`. A display verbatim such as

```
this is \display
```

can be made with `||...||`. If visible spaces are desired, use `@||...||` instead, which will produce

```
this␣is␣\display.
```

Every normal TEX character (except `|`) as well as all of the special characters can be used inside this environment, as shown in the following example:

```
!$@$#%^&*()-+=\ /.,;'`~th@<is$ ]\bye *is{ #a '`1\rho > *\tes<t* \undefined .??["
```

Normally, the pipe character `|` can only be used inside the display verbatim `||...||`. However, the macros `\makepipeother` and `\makepipeactive` turn the character `|` into an innocent character and into an active character, respectively. There is also the macro `\pipe` which expands to `\bgroup\string|\egroup`. These commands allow for use of the pipe literal `|` in horizontal mode. For example, both

```
{\tt th\pipe{}s is a pipe}
```

and

```
\makepipeother{\tt th|s is a pipe}\makepipeactive
```

will display `th|s is a pipe`. Of course, these macros cannot be used inside the `|...|` or `||...||` environments. In contrast, the character `@` is able to be used normally, provided that it is not proceeded by the pipe character; TEX will recognize the sequence `@|` as the beginning of a verbatim.

These verbatim environments obey spaces, lines, and blank lines; this allows for code insertion such as

```
void dfs(int p){
    if(o[p]) return;
    o[p]=1;

    c[p]=t++;
    for(int i=0; i<s[p].size(); ++i){
        dfs(s[p][i]);
    }
    Oops, the very long line of code is here, it does not fit into the line width unfor-█
tunately.
}
```

which will compile as expected.

As a backup (in case `|` is needed), the four commands `\bverb...\everb` and `\bverbatim...\everbatim` are provided as substitutes for `|...|` and `||...||`. Furthermore, the commands `\bverb` and `\bverbs` gobble the first space of their argument. However, these environments cannot be used with `@`; instead, use `\bverbs...\everbs` and `\bverbatims...\everbatims`. Use this command in front of either environment for visible spaces. For example, the code

```
\svs\bverb this is a \test\everb
```

will display `this␣is␣a␣\test`. These commands can also be used to display the `|` character: the code

```
\bverb th|s is a pipe\everb
```

will display `th|s is a pipe`, as expected. When the `\bverb` macro is executed, TEX turns every available character innocent until it scans the exact sequence `\everb`.

`laksdjflaksdjflkdsjf fthis is \undefined \bye sdflkjdf\ev@erb.`

asdfasdfasdfasdf

```
void dfs(int p){
    \undefined

}
```

asdfasdfasdfadsfadfasdf

```
void dfs(int p){
    \undefined -- --- ‘‘ ’’ ‘? ‘!

}
```

asdfasdfasdfasdf

this will display

$$sdfsdfs\rho$$

and this is the next