

# PlainT<sub>E</sub>X 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 T<sub>E</sub>X 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 back into an active character, respectively. This allows for use of the pipe literal in horizontal mode. There is also the macro `\pipe` which expands to

```
\bgroup\string|\egroup
```

displaying the literal `|`. These macros can be used to, for example, execute

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

or, as an alternative,

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

which will each display `th|s is a pipe`. Of course, these macros cannot be used inside the `|\...|` or `||\...||` environments. The character `@` is able to be used normally; T<sub>E</sub>X code such as

```
this is a t@st
@
```

```
this is a test@
\bye
```

will compile as expected. These verbatim environments also obey spaces, lines, and blank lines; this allows for indented 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]);
    }
}

```

which will also compile as expected.

As a backup (in case `|` is needed), the four commands `\bverb... \everb` and `\bverbatim... \verbatim` are provided as substitutes for `|...|` and `||...||`. However, these environments cannot be used with `@`; instead, the command `\svs` (short for `\showverbspaces`) is provided. 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.