```
        sub READ {
            my $self = shift;
            my $bufref = \$_[0];
            my(undef,$len,$offset) = @_;
            print "READ called, \$buf=$bufref, \$len=$len, \$offset=$offset";
            # add to $$bufref, set $len to number of characters read
            $len;
        }
```

READLINE this

This method will be called when the handle is read from via <HANDLE>. The method should return undef when there is no more data.

```
    sub READLINE { $r = shift; "READLINE called $$r times\n"; }
```

GETC this

This method will be called when the `getc` function is called.

```
    sub GETC { print "Don't GETC, Get Perl"; return "a"; }
```

CLOSE this

This method will be called when the handle is closed via the `close` function.

```
    sub CLOSE { print "CLOSE called.\n" }
```

UNTIE this

As with the other types of ties, this method will be called when `untie` happens. It may be appropriate to "auto CLOSE" when this occurs. See The `untie` Gotcha below.

DESTROY this

As with the other types of ties, this method will be called when the tied handle is about to be destroyed. This is useful for debugging and possibly cleaning up.

```
    sub DESTROY { print "</shout>\n" }
```

Here's how to use our little example:

```
    tie(*FOO,'Shout');
    print FOO "hello\n";
    $a = 4; $b = 6;
    print FOO $a, " plus ", $b, " equals ", $a + $b, "\n";
    print <FOO>;
```

45.2.5 UNTIE this

You can define for all tie types an UNTIE method that will be called at untie(). See The `untie` Gotcha below.