```
In[1]:= Needs["RG`BaseUtils`"]
```

assert

```
In[2]:= ? assert
```

assert[expr] evaluate expr with temporally enabled assertions

```
In[3]:= Assert[1 > 2]
Out[3]= Assert[1 > 2]

In[4]:= assert[1 > 2]
        Assert::asrtfl : Assertion 1 > 2 at line 34 in RG`BaseUtils` failed. >>
```

reload

```
In[5]:= ? reload
```

reload[context] remove definitions context`* and reload it

```
In[6]:= reload["RG`BaseUtils`", "verbose" → True]
        context: RG`BaseUtils`
        list of symbols to remove: {assert, modify, reload}
        list of loaded symbols: {assert, modify, reload}
```

OverTilde

```
In[7]:= ? OverTilde
```

OverTilde[func][args][expr] works as func[expr,
args] i.e. it creates operator OverTilde[func][args] for the first argument of func
>>

```
In[8]:= OverTilde[Sin][] [x]
Out[8]= Sin[x]
```

In[9]:= $\tilde{f}$[] [x]

Out[9]= f[x]

```
In[10]:= Labeled["test", Top][x]
         test
Out[10]=   x
```

modify

```
In[11]:= ? modify
```

modify[pattern, fs] create function to replace all matches of the
    pattern to results of consequent application of functions fs to these matches
modify[{x1, ...}, fs] create function for specific x1, ...

```
In[12]:= {1, a, b, 2} // modify[_Integer, Style[#, Red] &] // modify[{b}, Style[#, Brown] &]
Out[12]= {1, a, b, 2}

In[13]:= x[X[x[x]]] // modify[_Symbol, Style[Brown]] // modify[{X}, Style[Magenta]]
Out[13]= x[X[x[x]]]
```

In[14]:= $a^2 + 2\,a\,b + b^2 + c^2 + 2\,c\,d + d^2$ // modify[{Expand[(c + d) ^ 2]}, Factor, Style[Red]]

Out[14]= $a^2 + 2\,a\,b + b^2 + (c + d)^2$