Gekko 2.4 to 3.0 cheat sheet

Series	Gekko 2.4	Gekko 3.0
definition	series x = 1, 2, m, 3;	x = 1, 2, m(), 3;
local period	series <2020 2021> y = x;	y <2020 2021> = x;
blanks separation	series x = 1 2 m 3;	x = data('1 2 m 3');
dynamic series	series x = x[-1] + 1;	x <dyn> = x[-1] + 1;
element access	series y[2021] = x[2020];	y[2021] = x[2020];
name-composition	series y{i}a = x{i}b;	y{%i}a = x{%i}b;
name-composition	series y%i|a = x%i|b;	y%i|a = x%i|b;
list of names	series #m = 100;	{#m} = 100;
operators	series y ^ 5; series y % 5;	y ^= 5; y %= 5;
operators	series y + 5; series y +$ 5;	y += 5; y <keep=p> += 5;
operators	series y * 5; series y # 5;	y *= 5; y #= 5;
frequency symbol	collapse x.a = x.q;	collapse x!a = x!q;

Lists	Gekko 2.4	Gekko 3.0
list of strings	list m1 = a, b, c;	#m1 = a, b, c;
singleton list	list m1 = a;	#m1 = a,;
'funny' strings	list <direct> m2 = 007, 7z;	#m2 = 007, 7z;
mix strings and lists	list m3 = x, #m1, y, #m2;	#m3 = x, {#m1}, y, {#m2};
listfile f.lst out	list listfile f = a, b, c;	#(listfile f) = a, b, c;
listfile f.lst in	list m = #(listfile f);	#m = #(listfile f);
union	list m3 = #m1 &+ #m2;	#m3 = #m1 || #m2;
intersection	list m3 = #m1 &* #m2;	#m3 = #m1 && #m2;
difference	list m3 = #m1 &- #m2;	#m3 = #m1 - #m2;
concatenate	list m3 = #m1, #m2;	#m3 = #m1 + #m2;
prefix/suffix	list m = #m2 prefix = 'p';	#m = #m2.prefix('p');
remove element text	list m = #m2 strip = 'z';	#m = #m2.replaceinside('z', '');
sort/trim	list m = #m2 sort trim;	#m = #m2.sort().unique();
element access	string s = #m3[2];	%s = #m3[2];
print list elements	list ? #m;	prt #m;
print series	prt #m;	prt {#m};

Scalars	Gekko 2.4	Gekko 3.0
value definition	val v = 1.23;	%v = 1.23;
date definition	date d = 2021q1;	%d = 2021q1;
string definition	string s = 'xy5';	%s = 'xy5';
string concatenate	string s2 = 'a' + %s + 'b';	%s2 = 'a{%s}b'; //or like 2.4
print string	string ? %s;	prt %s;
print series	prt {%s};	prt {%s};

Wildcards	Gekko 2.4	Gekko 3.0
index with wildcard	index x* m;	index x* to #m;
index with range	index a..d m;	index a..d to #m;
search inside databank	list m2 = [x*], [a..d];	#m2 = ['x*'] + ['a'..'d'];
search inside list	list m2 = #m1[x*], #m1[a..d];	#m2 = #m1['x*'] + #m1['a'..'d'];
print matching series	prt [x*], [a..d];	prt {'x*'}, {'a'..'d'};

Loops	Gekko 2.4	Gekko 3.0

date loop
value loop
string loop
	for date d = %d1 to %d2; ...
	for val i = %i1 to %i2; ...
	for s = #m; ...
		for date %d = %d1 to %d2; ...
		for val %i = %i1 to %i2; ...
		for string %s = #m; ...

Matrix	Gekko 2.4	Gekko 3.0

matrix definition
element access
print matrix
print expression
	matrix m = [1, 2 || 3, 4];
	matrix m[1, 2] = #m[2, 1];
	matrix ? #m;
	show #m * #m;
		#m = [1, 2; 3, 4];
		#m[1, 2] = #m[2, 1];
		prt #m;
		prt #m * #m;

Miscellaneous	Gekko 2.4	Gekko 3.0

import for all periods
export for all periods
use {} for path parts
use {} for sum
value to survive read
list to survive read
	import <csv> data.csv;
	export <csv> data.csv;
	read c:\gekko\%path\data.gbk;
	series y = sum(#m);
	val v = 1.23;
	list m1 = a, b, c;
		import <csv all> data.csv;
		export <csv all> data.csv;
		read c:\gekko\{%path}\data.gbk;
		y = sum({#m});
		global:%v = 1.23;
		global:#m1 = a, b, c;

Model/sim example	Gekko 2.4	Gekko 3.0

The model/sim syntax is very similar. In 3.0, lists and scalars that are to survive READ statements must be placed in the Global databank. Series operators include a ’=’ symbol. And you must use {}-curlies when referring to strings and lists.	reset;
	mode sim;
	list fx = fxnx, fxqx;
	string path = 'c:\models';
	model %path\model2;
	read bank2;
	time 2021 2025;
	sim;
	clone;
	series tg + 0.01;
	sim;
	mulprt #fx;
	read bank2;
	endo tg;
	endo fy;
	sim;
	mulprt tg, fy, #fx;
		reset;
		mode sim;
		global:#fx = fxnx, fxqx;
		global:%path = 'c:\models';
		model {%path}\model2;
		read bank2;
		time 2021 2025;
		sim;
		clone;
		tg += 0.01;
		sim;
		mulprt {#fx};
		read bank2;
		endo tg;
		endo fy;
		sim;
		mulprt tg, fy, {#fx};

About {}-curlies in Gekko 3.0

In Gekko 3.0, it is more often required to enclose a string %s or a list of strings #m in curlies, that is, {%s} or {#m}. This applies to the following commands: assignments, analyze, checkoff, collapse, compare, copy, disp, doc, endo, exo, export, findmissingdata, import, interpolate, itershow, ols, prt/mulprt/plot/sheet, read, rebase, rename, smooth, splice, truncate, write, x12a, where assignments refer to series/val/string/list etc. statements. Consider these two lines in Gekko 3.0:

```
  delete {%s}, {#m};
  delete %s, #m;
```

If the string %s contains 'x1' and the list #m contains the strings 'x2', 'x3', 'x4', the first statement deletes the series x1, x2, x3, x4 (4 in all). In the next statement, it is the string and list objects %s and #m themselves that are deleted (2 in all). Understanding this conceptual difference is important!

Own notes: