SLiM Reference Sheet

11 May 2020

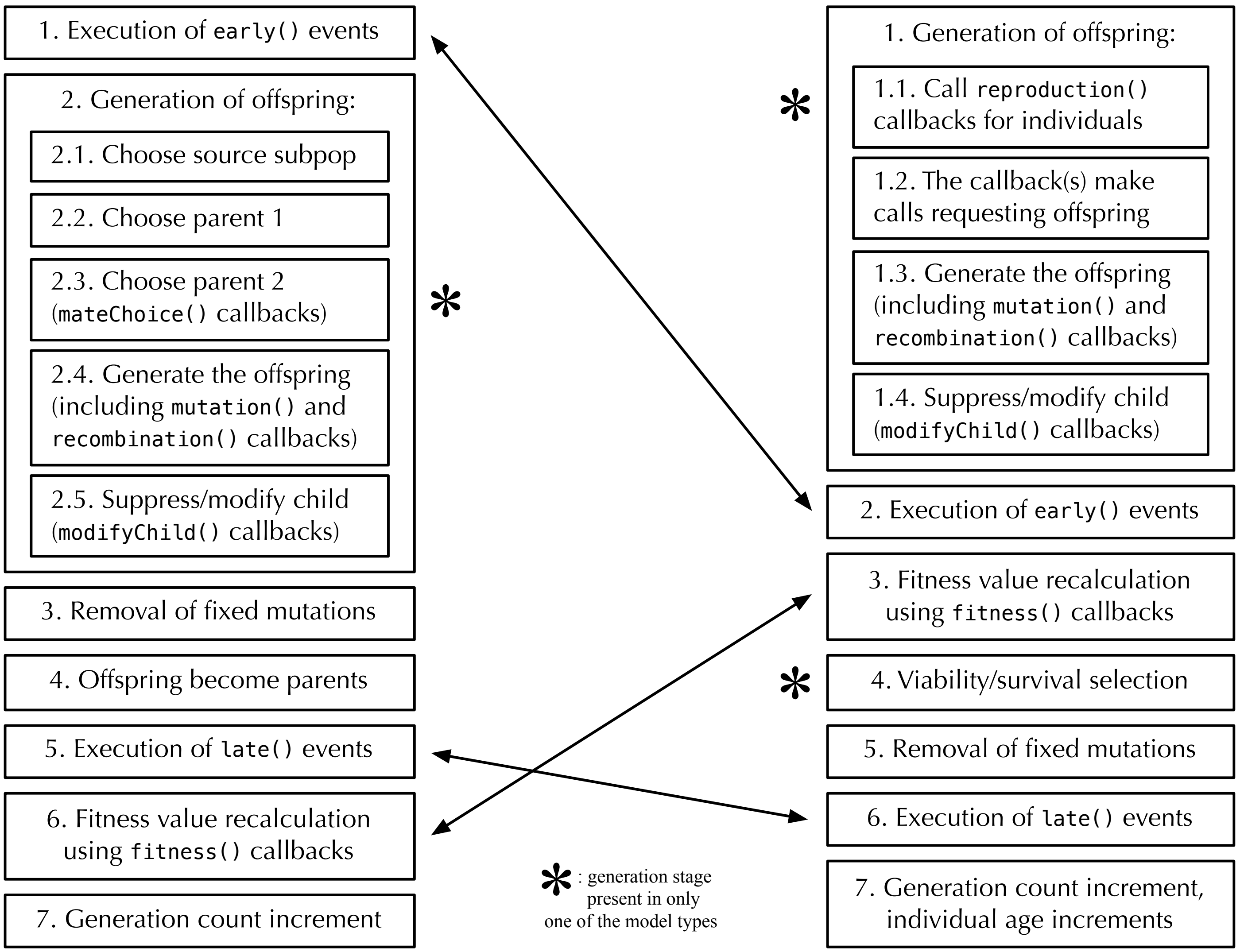
```
slim -version | -usage | -testEidos | -testSLiM |
  [-seed <seed>] [-time] [-mem] [-Memhist] [-long [<l>]] [-x] [-define <def>] <script file>
```

Types: `N:NULL, l:logical, i:integer, f:float, n:numeric, s:string, o<X>:object of class X`

`(ret-type)functionName(params) { ... }`	user-defined function
`[<id>] initialize() { ... }`	initialize() callback
`[<id>] [gen1 [: gen2]] [early()] { ... }`	early() Eidos event
`[<id>] [gen1 [: gen2]] late() { ... }`	late() Eidos event
`[<id>] [gen1 [: gen2]] fitness(<mutTypeId> [, <subpopId>]) { ... }`	fitness() callback
`[<id>] [gen1 [: gen2]] interaction(<intTypeId> [, <subpopId>]) { ... }`	interaction() callback
`[<id>] [gen1 [: gen2]] mateChoice([<subpopId>]) { ... } (WF)`	mateChoice() callback
`[<id>] [gen1 [: gen2]] modifyChild([<subpopId>]) { ... }`	modifyChild() callback
`[<id>] [gen1 [: gen2]] recombination([<subpopId>]) { ... }`	recombination() callback
`[<id>] [gen1 [: gen2]] mutation([<mutTypeId> [, <subpopId>]]) { ... }`	mutation() callback
`[<id>] [gen1 [: gen2]] reproduction([<subpopId> [, <sex>]]) ... (nonWF)`	reproduction() callback

The sequence of events within one generation in WF models.

The sequence of events within one generation in nonWF models.



1. Execution of `early()` events

2. Generation of offspring:

2.1. Choose source subpop

2.2. Choose parent 1

2.3. Choose parent 2 (`mateChoice()` callbacks)

2.4. Generate the offspring (including `mutation()` and `recombination()` callbacks)

2.5. Suppress/modify child (`modifyChild()` callbacks)

3. Removal of fixed mutations

4. Offspring become parents

5. Execution of `late()` events

6. Fitness value recalculation using `fitness()` callbacks

7. Generation count increment

1. Generation of offspring:

1.1. Call `reproduction()` callbacks for individuals

1.2. The callback(s) make calls requesting offspring

1.3. Generate the offspring (including `mutation()` and `recombination()` callbacks)

1.4. Suppress/modify child (`modifyChild()` callbacks)

2. Execution of `early()` events

3. Fitness value recalculation using `fitness()` callbacks

4. Viability/survival selection

5. Removal of fixed mutations

6. Execution of `late()` events

7. Generation count increment, individual age increments

* : generation stage present in only one of the model types

Initialization functions (callable only from initialize() callbacks):

```
(integer$)initializeAncestralNucleotides(is sequence) (nuc)
(void)initializeGeneConversion(n$ nonCrossoverFraction, n$ meanLength,
      n$ simpleConversionFraction, [n$ bias])
(o<GElement>)initializeGenomicElement(io<GEType> genomicElementType, i start, i end)
(o<GEType>$)initializeGenomicElementType(is$ id, io<MutType> mutationTypes, n proportions,
      [Nf mutationMatrix])
(void)initializeHotspotMap(numeric multipliers, [Ni ends], [s$ sex]) (nuc)
(o<IntType>$)initializeInteractionType(is$ id, s$ spatiality, [l$ reciprocal],
      [n$ maxDistance], [s$ sexSegregation])
(void)initializeMutationRate(n rates, [Ni ends], [s$ sex])
(o<MutType>$)initializeMutationType(is$ id, n$ dominanceCoeff, s$ distributionType, ...)
(o<MutType>$)initializeMutationTypeNuc(is$ id, n$ dominanceCoeff, s$ distributType, ...) (nuc)
(void)initializeRecombinationRate(n rates, [Ni ends], [s$ sex])
(void)initializeSex(s$ chromosomeType, [n$ xDominanceCoeff])
(void)initializeSLiMModelType(s$ modelType)
(void)initializeSLiMOptions([l$ keepPedigrees], [s$ dimensionality], [s$ periodicity],
      [i$ mutationRuns], [l$ preventIncidentalSelfing], [l$ nucleotideBased])
(void)initializeTreeSeq([l$ recordMutations], [Nif$ simplificationRatio],
      [Ni$ simplificationInterval], [l$ checkCoalescence], [l$ runCrosschecks])
```

fitness():

```
  mut (o<Mutation>$)
  homozygous (l$)
  relFitness (f$)
  individual (o<Ind>$)
  genome1 (o<Genome>$)
  genome2 (o<Genome>$)
  subpop (o<Subpop>$)
```

reproduction(): (nonWF)

```
  individual (o<Ind>$)
  genome1 (o<Genome>$)
  genome2 (o<Genome>$)
  subpop (o<Subpop>$)
```

Individual (Ind):

```
age <–> (i$) (nonWF)
color <–> (s$)
fitnessScaling <–> (f$)
genomes => (o<Genome>)
genome1 => (o<Genome>$)
genome2 => (o<Genome>$)
index => (i$)
migrant => (l$)
pedigreeID => (i$)
pedigreeParentIDs => (i)
pedigreeGrandparentIDs => (i)
sex => (s$)
spatialPosition => (f)
subpopulation => (o<Subpop>$)
tag <–> (i$)
tagF <–> (f$)
uniqueMutations => (o<Mut>)
x <–> (f$)
y <–> (f$)
z <–> (f$)
– (l)containsMutations(o<Mut> mutations)
– (i$)countOfMutationsOfType(io<MutType>$ mutType)
– (+)getValue(s$ key)
– (f)relatedness(o<Ind> individuals)
+ (void)setSpatialPosition(f position)
– (void)setValue(s$ key, + value)
– (f$)sumOfMutationsOfType(io<MutType>$ mutType)
– (o<Mut>)uniqueMutationsOfType(io<MutType>$ mutType)
```

modifyChild():

```
  child (o<Ind>$)
  childGenome1 (o<Genome>$)
  childGenome2 (o<Genome>$)
  childIsFemale (l$)
  parent1 (o<Ind>$)
  parent1Genome1 (o<Genome>$)
  parent1Genome2 (o<Genome>$)
  isCloning (l$)
  isSelfing (l$)
  parent2 (o<Ind>$)
  parent2Genome1 (o<Genome>$)
  parent2Genome2 (o<Genome>$)
  subpop (o<Subpop>$)
  sourceSubpop (o<Subpop>$)
```

mateChoice(): (WF)

```
  individual (o<Ind>$)
  genome1 (o<Genome>$)
  genome2 (o<Genome>$)
  subpop (o<Subpop>$)
  sourceSubpop (o<Subpop>$)
  weights (f)
```

recombination():

```
  individual (o<Ind>$)
  genome1 (o<Genome>$)
  genome2 (o<Genome>$)
  subpop (o<Subpop>$)
  breakpoints (i)
  ~~gcStarts (i)~~
  ~~gcEnds (i)~~
```

mutation():

```
  mut (o<Mutation>$)
  genome (o<Genome>$)
  element (o<GElement>$)
  originalNuc (i$)
  parent (o<Ind>$)
  subpop (o<Subpop>$)
```

interaction():

```
  distance (f$)
  strength (f$)
  exerter (o<Ind>$)
  receiver (o<Ind>$)
  subpop (o<Subpop>$)
```

SLiM globals:

```
  sim (o<SLiMSim>$)
  g1, ... (o<GEType>$)
  i1, ... (o<IntType>$)
  m1, ... (o<MutType>$)
  p1, ... (o<Subpop>$)
  s1, ... (o<SEBlock>$)
  self (o<SEBlock>$)
```

SLiMgui quick help:
opt-click on keyword

Code completion:
escape (ESC) or
cmd-period (⌘.)

SLiMSim:

```
chromosome => (o<Chromosome>$)
chromosomeType => (s$)
dimensionality => (s$)
dominanceCoeffX <–> (f$)
generation <–> (i$)
genomicElementTypes => (o<GEType>)
~~inSLiMgui => (l$)~~ (deprecated in SLiM 3.2.1)
interactionTypes => (o<IntType>)
modelType => (s$)
mutationTypes => (o<MutType>)
mutations => (o<Mut>)
nucleotideBased => (l$) (nuc)
periodicity => (string$)
scriptBlocks => (o<SEBlock>)
sexEnabled => (l$)
subpopulations => (o<Subpop>)
substitutions => (o<Substitution>)
tag <–> (i$)
```

```
– (o<Subpop>$)addSubpop(is$ subpopID, i$ size, [f$ sexRatio])
– (o<Subpop>$)addSubpopSplit(is$ subpopID, i$ size, io<Subpop>$ sourceSubpop,
      [f$ sexRatio]) (WF)
– (i$)countOfMutationsOfType(io<MutType>$ mutType)
– (void)deregisterScriptBlock(io<SEBlock> scriptBlocks)
– (+)getValue(s$ key)
– (i)mutationCounts(No<Subpop> subpops, [No<Mut> mutations])
– (f)mutationFrequencies(No<Subpop> subpops, [No<Mut> mutations])
– (o<Mut>)mutationsOfType(io<MutType>$ mutType)
– (void)outputFixedMutations([Ns$ filePath], [l$ append])
– (void)outputFull([Ns$ filePath], [l$ binary], [l$ append], [l$ spatialPositions], [l$ ages])
– (void)outputMutations(o<Mut> mutations, [Ns$ filePath], [l$ append])
– (void)outputUsage(void)
– (i$)readFromPopulationFile(s$ filePath)
– (void)recalculateFitness([Ni$ generation])
– (o<SEBlock>$)register[Early/Late]Event(Nis$ id, s$ source, [Ni$ start], [Ni$ end])
– (o<SEBlock>$)registerFitnessCallback(Nis$ id, s$ source, Nio<MutType>$ mutType,
      [Nio<Subpop>$ subpop], [Ni$ start], [Ni$ end])
– (o<SEBlock>$)registerInteractionCallback(Nis$ id, s$ source, io<IntType>$ intType,
      [Nio<Subpop>$ subpop], [Ni$ start], [Ni$ end])
– (o<SEBlock>$)register[MateChoice (WF)/ModifyChild/Recombination]Callback(Nis$ id, s$ source,
      [Nio<Subpop>$ subpop], [Ni$ start], [Ni$ end])
– (o<SEBlock>$)registerMutationCallback(Nis$ id, s$ source, [Nio<MutType>$ mutType],
      [Nio<Subpop>$ subpop], [Ni$ start], [Ni$ end])
– (o<SEBlock>$)registerReproductionCallback(Nis$ id, s$ source, [Nio<Subpop>$ subpop],
      [Ns$ sex], [Ni$ start], [Ni$ end]) (nonWF)
– (o<SEBlock>$)rescheduleScriptBlock(o<SEBlock>$ block, [Ni$ start], [Ni$ end],
      [Ni generations])
– (void)setValue(s$ key, + value)
– (void)simulationFinished(void)
– (l$)treeSeqCoalesced(void)
– (void)treeSeqOutput(s$ path, [l$ simplify], [l$ includeModel])
– (void)treeSeqRememberIndividuals(o<Ind> individuals)
– (void)treeSeqSimplify(void)
```

SLiM built-in functions:

```
– (s)codonsToAminoAcids(i codons, [l$ long], [l$ paste]) (nuc)
– (is)codonsToNucleotides(i codons, [s$ format]) (nuc)
– (f)mm16To256(f mutationMatrix) (nuc)
– (f)mmJukesCantor(f$ alpha) (nuc)
– (f)mmKimura(f$ alpha, f$ beta) (nuc)
– (i)nucleotideCounts(is sequence) (nuc)
– (f)nucleotideFrequencies(is sequence) (nuc)
– (i)nucleotidesToCodons(is sequence) (nuc)
– (is)randomNucleotides(i$ length, [Nif basis], [s$ format]) (nuc)
```

Mutation (Mut):

```
id => (i$)
mutationType => (o<MutType>$)
nucleotide <–> (s$) (nuc)
nucleotideValue <–> (i$) (nuc)
originGeneration => (i$)
position => (i$)
selectionCoeff => (f$)
subpopID <–> (i$)
tag <–> (i$)
– (+)getValue(s$ key)
– (void)setMutationType(io<MutType>$ mutType)
– (void)setSelectionCoeff(f$ selectionCoeff)
– (void)setValue(s$ key, + value)
```

Substitution:

```
id => (i$)
fixationTime => (i$)
mutationType => (o<MutType>$)
nucleotide <–> (s$) (nuc)
nucleotideValue <–> (i$) (nuc)
originGeneration => (i$)
position => (i$)
selectionCoeff => (f$)
subpopID <–> (i$)
tag <–> (i$)
– (+)getValue(s$ key)
– (void)setValue(s$ key, + value)
```

Subpopulation (Subpop):

```
cloningRate => (f) (WF)
firstMaleIndex => (i$)
fitnessScaling <–> (f$)
genomes => (o<Genome>)
id => (i$)
immigrantSubpopFractions => (f) (WF)
immigrantSubpopIDs => (i) (WF)
individualCount => (i$)
individuals => (o<Ind>)
selfingRate => (f$) (WF)
sexRatio => (f$) (WF)
spatialBounds => (f)
tag <–> (i$)
– (No<Ind>$)addCloned(o<Ind>$ parent) (nonWF)
– (No<Ind>$)addCrossed(o<Ind>$ parent1, o<Ind>$ parent2, [Nfs$ sex]) (nonWF)
– (No<Ind>$)addEmpty([Nfs$ sex]) (nonWF)
– (No<Ind>$)addRecombinant(No<Genome>$ strand1, No<Genome>$ strand2, Ni breaks1,
      No<Genome>$ strand3, No<Genome>$ strand4, Ni breaks2, [Nfs$ sex]) (nonWF)
– (No<Ind>$)addSelfed(o<Ind>$ parent) (nonWF)
– (f)cachedFitness(Ni indices)
– (void)configureDisplay([Nf center], [Nf$ scale], [Ns$ color])
– (void)defineSpatialMap(s$ name, s$ spatiality, Ni gridSize, n values, [l$ interpolate],
      [Nif valueRange], [Ns colors])
– (+)getValue(s$ key)
– (void)outputMSSample(i$ sampleSize, [l$ replace], [s$ requestedSex],
      [Ns$ filePath], [l$ append], [l$ filterMonomorphic])
– (void)outputSample(i$ sampleSize, [l$ replace], [s$ requestedSex],
      [Ns$ filePath], [l$ append])
– (void)outputVCFSample(i$ sampleSize, [l$ replace], [s$ requestedSex],
      [l$ outputMultiallelics], [Ns$ filePath], [l$ append])
– (l)pointInBounds(f point)
– (f)point[Periodic|Reflected|Stopped](f point)
– (f)pointUniform([i$ n])
– (void)removeSubpopulation(void) (nonWF)
– (void)sampleIndividuals(i$ size, [l$ replace], [No<Ind>$ exclude], [Ns$ sex], [Ni$ tag],
      [Ni$ minAge], [Ni$ maxAge], [Nl$ migrant])
– (void)setCloningRate(n rate) (WF)
– (void)setMigrationRates(io<Subpop> sourceSubpops, n rates) (WF)
– (void)setSelfingRate(n$ rate) (WF)
– (void)setSexRatio(f$ sexRatio) (WF)
– (void)setSpatialBounds(n bounds)
– (void)setSubpopulationSize(i$ size) (WF)
– (void)setValue(s$ key, + value)
– (s)spatialMapColor(s$ name, n value)
– (f)spatialMapValue(s$ name, f point)
– (void)subsetIndividuals([No<Ind>$ exclude], [Ns$ sex], [Ni$ tag], [Ni$ minAge],
      [Ni$ maxAge], [Nl$ migrant])
– (void)takeMigrants(o<Ind> migrants) (nonWF)
```

Fitness effects of mutations:

no mutation present	1
heterozygote	$1 + h*s$
homozygote	$1 + s$

s = `mut.selectionCoeff`
h = `mutType.dominanceCoeff`

Genome:

```
genomePedigreeID => (i$)
genomeType => (s$)
individual => (o<Ind>$)
isNullGenome => (l$)
mutations => (o<Mut>)
tag <–> (i$)
```

```
+ (void)addMutations(o<Mut> mutations)
+ (o<Mut>)addNewDrawnMutation(io<MutType> mutationType, i position, [Ni originGeneration],
      [Nio<Subpop> originSubpop], [Nis nucleotide])
+ (o<Mut>)addNewMutation(io<MutType> mutationType, n selectionCoeff, i position,
      [Ni originGeneration], [Nio<Subpop> originSubpop], [Nis nucleotide])
– (l$)containsMarkerMutation(io<MutType>$ mutType, i$ position, [l$ returnMutation])
– (l)containsMutations(o<Mut> mutations)
– (i$)countOfMutationsOfType(io<MutType>$ mutType)
– (o<Mut>)mutationsOfType(io<MutType>$ mutType)
– (is)nucleotides([Ni$ start], [Ni$ end], [s$ format])
+ (void)output([Ns$ filePath], [l$ append])
+ (void)outputMS([Ns$ filePath], [l$ append], [l$ filterMonomorphic])
+ (void)outputVCF([Ns$ filePath], [l$ outputMultiallelics], [l$ append])
– (i)positionsOfMutationsOfType(io<MutType>$ mutType)
+ (o<Mut>)readFromMS(s$ filePath, io<MutType>$ mutationType)
+ (o<Mut>)readFromVCF(s$ filePath, [Nio<MutType>$ mutationType])
+ (void)removeMutations([No<Mut> mutations], [l$ substitute])
– (f$)sumOfMutationsOfType(io<MutType>$ mutType)
```

Chromosome:

```
colorSubstitution <–> (s$)
geneConversionEnabled => (l$)
geneConversionGCBias => (f$)
geneConversionNonCrossoverFraction => (f$)
geneConversionMeanLength => (f$)
geneConversionSimpleConversionFraction => (f$)
genomicElements => (o<GElement>)
hotspotEndPositions[F|M] => (i) (nuc)
hotspotMultipliers[F|M] => (f) (nuc)
lastPosition => (i$)
mutationEndPositions[F|M] => (i)
mutationRates[F|M] => (f)
overallMutationRate[F|M] => (f$)
overallRecombinationRate[F|M] => (f$)
recombinationEndPositions[F|M] => (i)
recombinationRates[F|M] => (f)
tag <–> (i$)
```

```
– (is)ancestralNucleotides([Ni$ start], [Ni$ end], [s$ format]) (nuc)
– (integer)drawBreakpoints([No<Ind>$ parent], [Ni$ n])
– (integer$)setAncestralNucleotides(is sequence) (nuc)
– (void)setGeneConversion(n$ nonCrossoverFraction, n$ meanLength,
      n$ simpleConversionFraction, [n$ bias])
– (void)setHotspotMap(n multipliers, [Ni ends], [s$ sex]) (nuc)
– (void)setMutationRate(n rates, [Ni ends], [s$ sex])
– (void)setRecombinationRate(n rates, [Ni ends], [s$ sex])
```

SLiMEidosBlock (SEBlock):

```
active <–> (i$)
end => (i$)
id => (i$)
source => (s$)
start => (i$)
tag <–> (i$)
type => (s$)
```

GenomicElementType (GEType):

```
color <–> (s$)
id => (i$)
mutationFractions => (f)
mutationMatrix => (f) (nuc)
mutationTypes => (o<MutType>)
tag <–> (i$)
```

```
– (+)getValue(s$ key)
– (void)setMutationFractions(io<MutType> mutationTypes, n proportions)
– (void)setMutationMatrix(f mutationMatrix) (nuc)
– (void)setValue(s$ key, + value)
```

GenomicElement (GElement):

```
endPosition => (i$)
genomicElementType => (o<GEType>$)
startPosition => (i$)
tag <–> (i$)
```

```
– (void)setGenomicElementType(io<GEType>$ genomicElementType)
```

MutationType (MutType):

```
color <–> (s$)
colorSubstitution <–> (s$)
convertToSubstitution <–> (l$)
distributionParams => (f)
distributionType => (s$)
dominanceCoeff <–> (f$)
id => (i$)
mutationStackGroup <–> (i$)
mutationStackPolicy <–> (s$)
nucleotideBased => (l$) (nuc)
tag <–> (i$)
```

```
– (float)drawSelectionCoefficient(i$ n)
– (+)getValue(s$ key)
– (void)setDistribution(s$ distType, ...)
– (void)setValue(s$ key, + value)
```

InteractionType (IntType):

```
id => (i$)
maxDistance <–> (f$)
reciprocal => (l$)
sexSegregation => (s$)
spatiality => (s$)
tag <–> (i$)
```

```
– (f)distance(o<Ind> individuals1, [No<Ind> individuals2])
– (f)distanceToPoint(o<Ind> individuals1, f point)
– (o<Ind>)drawByStrength(o<Ind>$ individual, [i$ count])
– (void)evaluate([No<Subpop> subpops], [l$ immediate])
– (+)getValue(s$ key)
– (i)interactingNeighborCount(o<Ind> individuals)
– (f)interactionDistance(o<Ind> individuals1, [No<Ind> individuals2])
– (o<Ind>)nearestInteractingNeighbors(o<Ind>$ individual, [i$ count])
– (o<Ind>)nearestNeighbors(o<Ind>$ individual, [i$ count])
– (o<Ind>)nearestNeighborsOfPoint(o<Subpop>$ subpop, f point, [i$ count])
– (void)setInteractionFunction(s$ functionType, ...)
– (void)setValue(s$ key, + value)
– (f)strength(o<Ind>$ receiver, [No<Ind> exerters])
– (f)totalOfNeighborStrengths(o<Ind> individuals)
– (void)unevaluate(void)
```

SLiMgui: `(in SLiMgui only)`

```
pid => (i$)
```

```
– (void)openDocument(s$ filePath)
– (void)pauseExecution(void)
```