
Phenotype Syntax

Revision \$Id\$

Table of Contents

Scope	1
Grammar	2
Tokenization rules	2
Informal Description	2
Elements	3
Type references	4
Examples	4
Example	5
Example	5
Changes	6
version 0.11	6
version 0.10	6

Chris Mungall 2006

Abstract

Pheno-syntax is a formal syntax for representing phenotypes. The syntax is designed to be compact, human-readable and writable. Use of pheno-xml over pheno-syntax is encouraged, but a concrete syntax offers advantages over an XML representation:

- human readability (as opposed to human legibility in XML)
- human writeability (without use of a specialised XML editor)

The syntax was designed in response to curator need after the May 2006 phenotype ontology meeting in Hinxton. XML was perceived to be unsuitable for certain users needs.

The syntax has a formal grammar, described below. The grammar indicates which tags are allowed in which contexts, and the cardinality of those tags. These should be in agreement with the XML.

This document primarily describes the syntax. Some mention is made of the interpretation of the syntax in terms of real-world entities. However, care has been taken not to be too redundant with the pheno-xml documentation. In addition, the documentation regarding the precise logical interpretation of pheno-xml and pheno-syntax will reside elsewhere (this may take the form of a transformation to either/both description logic or full first order logic); the documentation here is intended for both savvy users (eg advanced curators) and programmers.

Scope

Unlike pheno-xml, pheno-syntax does not attempt to describe genotypes or specific manifestations of phenotypes. Rather, pheno-syntax is intended to be embedded into existing file formats. The details of that embedding are not part of pheno-syntax.

Pheno-syntax may be slightly less expressive than pheno-xml in some places. There are tradeoffs to keep the language simple.

Grammar

```
phenotype ::= phenotype_typeof? phenotype_character*
```

```
phenotype_character ::= description? expressivity? bearer quality*
```

```
expressivity ::= 'Expressivity=' float '%'
```

```
bearer ::= 'E=' typeof
```

```
quality ::= 'Q=' typeof count? related_entity* on_condition* in_comparison
```

```
count ::= 'C=' integer
```

```
related_entity ::= 'E2=' typeof
```

```
on_condition ::= 'On=' typeof
```

```
in_comparison_to ::= 'Compar=' typeof comparison_target?
```

```
measurement ::= 'M=' float unit
```

```
temporal_qualifier ::= 'T=' qualifier
```

```
modifier ::= 'Tag=' id
```

```
phenotype_typeof ::= 'P=' typeof
```

```
comparison_target ::= '{' relation? typeof '}'
```

```
description ::= 'Desc=' ''' text '''
```

```
relation ::= typeof
```

```
typeof ::= id conjunction*
```

```
conjunction ::= '^' qualifier
```

```
qualifier ::= relation '(' typeof ')'
```

```
id ::= prefix ':' local_id
```

```
prefix ::= word
```

```
local_id ::= word
```

```
unit ::= word
```

Tokenization rules

The syntax is not newline or formatting sensitive. All whitespace is treated equally. Can be arranged/formatted as desired

Informal Description

Pheno-syntax consists of a collection of tag-values, which look something like this:

E= GO:0012345 Q= PATO:0000001

The syntax is whitespace-neutral, so the above can be written as

```
E= GO:0012345
  Q= PATO:0000001
```

However, there should be no space between the tag name and the =. Also, tag names are case sensitive

A phenotype can be broken down into an `_optional_` text description followed by zero or more phenotype characters. Descriptions use the `Desc` tag, and the description must be enclosed in double quotes

```
Desc= "Heterozygous flies have very short and highly branched arista
      laterals."
```

A phenotype character consists of a single `E=` (bearer) tag, and zero or more `Q=` (quality/attribute) tags. Each `Q=` tag can be followed by additional optional tags

Note that no additional separators are required - the "nesting" is completely unambiguous and is specified by the formal grammar, above. In the following example whitespace is used to indicate the structure of the annotation. However, a parser ignores the whitespace and can unambiguously retrieve the structure

```
Desc= "foo"
E= GO:0012345
  Q= PATO:0000001
  Q= PATO:0000002
    E2= CHEBI:345
E= GO:0099999
  Q= PATO:0000003
  Q= PATO:0000004
```

Elements

Element	Parent	Description
E		The "bearer" entity type
		- can be from any ontology of processes or continuants
		for example GO, CL, anatomy, ChEBI
Q	E	The quality type, which <i>inheres_in</i> the entity
		- a PATO ID
		* multiple Qs can be specified for any E
C	Q	Count - an integer
		Only for certain PATO types like <i>supernumerary</i>
		* zero-or-one C per Q
E2	Q	Related entity type -
		For use with relational qualities such as "sensitivity"
		or "lacking part"

Element	Parent	Description
		* zero or more per Q (more than 1 will be rare)
On	Q	A condition which must be satisfied for the phenotype to be expressed; may be a term from an environment ontology
		* zero or more per Q
Compar	Q	In comparison to. Can specify an additional quality type here. Optionally followed by a typeref from a taxonomy (the target org type)
		* zero or more per Q
M	Q	Measurement. A number followed by a unit name
		* zero or more per Q
T	Q	Temporal qualifier
		Takes the form of a qualifier (see below)
		for example: <i>during(GO:cell_cycle)</i>
		* zero or more per Q
Tag	Q	Modifier ID. Taken from PATO
		eg absent, abnormal

Type references

A type reference is typically an OBO ID which references a representation of a type, drawn from some ontology. These are specified in normal OBO ID form, eg *GO:0000001*

If sufficiently expressive terms have not been pre-coordinated within the ontology, we can post-coordinate them in pheno-syntax. This is done by specifying the *core* terms (aka *generic/genus* term), followed by one or more *qualifiers* (aka *discriminating characteristics* or *differentiae*). Each is separated with the ^ symbol.

For example - if CL:123 represents the type "sperm" and GO:234 represents the type "nucleus" we can write:

```
GO:234^part_of(CL:123)
```

to represent "a nucleus which is part of a sperm cell"

Order is important.

Relations are assumed to come from the OBO relation ontology. If other relations are required, a full OBO ID can be specified here (although use of relations outside OBO REL is highly discouraged)

Examples

In the examples that follow we do not use actual numeric OBO IDs - we "fake" the IDs for the sake of clarity

Example

"Heterozygous flies have very short and highly branched arista laterals."

"The decreased length of the laterals is associated with a decreased rate of elongation and premature cessation of elongation."

```
E= FBbt:arista_lateral
  Q= PATO:short
E= GO:elongation_of_arista_lateral
  Q= PATO:low_rate Q= PATO:premature_cessation
```

What if GO did not contain the pre-coordinated term "elongation of arista lateral" (GO:0035016)? We could post-coordinate a similar type, in this way:

```
E= GO:growth ^ has_participant(FBbt:arista_lateral)
  Q= PATO:low_rate
  Q= PATO:premature_cessation
```

"There is no delay in lateral outgrowth."

```
E=FBbt:arista_lateral Q=PATO:width Tag=PATO:normal
```

"Initially a single lateral is formed, but it is thicker than normal and sometimes individual bundles of actin filaments can be seen. The laterals split near the distal tip as they elongate and at later stages they appear to be split over more of their length."

```
E=ASPO:distal_region ^ part_of(FBbt:arista_lateral)
  Q=PATO:branched
```

"The shape of the laterals in cross section is often irregular, in contrast to the circular wild-type laterals."

```
E= ASPO:cross_section ^ part_of(FBbt:arista_lateral)
  Q= PATO:irregular_shape
  Tag= PATO:abnormal
  Compar= PATO:circular {NCBITaxon:7227}
```

"The actin bundles are irregular in shape and extend further into the centre of the lateral than normal, although most are still associated with the plasma membrane. The actin filaments occupy more of the lateral cross section than normal, although they appear less tightly packed than wild type. The density of microtubules is significantly reduced and their distribution is more irregular compared to wild type."

```
E=GO:actin_filament_organisation
  Q=PATO:packed
```

Example

"In uncrowded conditions 48% eclose as adults although development is delayed by 1-2 days. Wings are usually unexpanded and many flies are found stuck in the food. The remaining 52% have difficulty eclosing and die as pharate adults. Transheterozygotes with Ca- Δ 1D[X7] are almost all able to escape

the puparium but few transheterozygotes with Ca- Δ ;1D[X10] succeed."

E= GO:eclosion
Q= PATO:arrested
E= FBbt:whole_organism
Q= PATO:lethal
T= during(FBdv:adult_stage)

Notes: to annotate transheterozygotes a separate phenotype entry would be used, and this phenotype entry would be associated with Ca- Δ ;1D[X7]. The annotation of the genotype is outwith the scope of pheno-syntax

"A few homozygotes exhibit neurons in the longitudinal tracts that appear to stall at some of the commissures, forming nodular growths. Motoneurons in the SNb branch of the developing PNS also show stalling at stage 17. The defects may be due to a second lesion on the chromosome or generated by altered channel properties caused by the missense mutation."

1. ???

E=
GO:development has_participant(FBbt:motor_neuron₁ has_location(FBbt:abdominal_posterior_fascicle))
Q= PATO:stalled T= during(FBdv:stage_17)

Changes

version 0.11

Changed cardinality of Compar tag to zero-or-more

version 0.10

original release