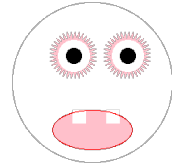


Documentation for the `potatohead.rkt` teachpack

This teachpack is intended to provide a gentle introduction to creating images. Information on how to install teachpacks can be found on the “DrRacket & Teachpacks” page of the course website.

We introduce a new type of data, a *PH* (a short form for “potatohead”), as well as functions that can be used to consume attributes and produce a potatohead, consume a potatohead and produce one of its attributes, consume a potatohead and produce a potatohead formed by changing an attribute, and display a potatohead.



1 Potatohead images

The following functions create a potatohead and display it. The colours are chosen from those used in the `world.rkt` teachpack. Some are listed in the documentation for the teachpack, linked off the “DrRacket & Teachpacks” page of the course website. You do not need to read the entire documentation for `world.rkt` to be able to use potatoheads.

Eye types are `'circle`, `'x`, `'star`, `'lashes`, and `'line`, and mouth types are `'oh`, `'happy`, `'tooth`, and `'line`.

Note: left and right correspond to the left and right of the image as we see it, not the left and right of a potatohead.

```
;; (create-ph head-colour radius l-eye-type l-eye-colour r-eye-type r-eye-colour
;;           mouth-type mouth-colour)
;; produces a PH with head colour head-colour, head size radius,
;; left eye of type l-eye-type and colour l-eye-colour,
;; right eye of type r-eye-type and colour r-eye-colour,
;; and mouth of type mouth-type and colour mouth-colour;
;; the position is automatically set to 50 50
;; create-ph: Sym Nat
;;           (anyof 'circle 'x 'star 'lashes 'line) Sym
;;           (anyof 'circle 'x 'star 'lashes 'line) Sym
;;           (anyof 'oh 'happy 'tooth 'line) Sym → PH
;; requires: head-colour, l-eye-colour, r-eye-colour, and mouth-colour
;;           are all colours recognized by world.rkt
(create-ph head-colour radius l-eye-type l-eye-colour r-eye-type r-eye-colour
           mouth-type mouth-colour)

;; (draw-ph aph) displays an image of the given potatohead aph.
;; draw-ph: PH → Image
(draw-ph aph)
```

For example, the following function applications will create a potatohead and display it:

```
(define myph (create-ph 'blue 50 'x 'red 'circle 'orange 'oh 'green))
(draw-ph myph)
```

Each of the following function applications can be used to determine an attribute of a potatohead *aph*. The query about size produces a number; each of the others produces a symbol.

(*what-head-colour aph*)
(*what-head-size aph*)
(*what-l-eye-type aph*)
(*what-l-eye-colour aph*)
(*what-r-eye-type aph*)
(*what-r-eye-colour aph*)
(*what-mouth-type aph*)
(*what-mouth-colour aph*)

The next functions are used to form a new potatohead based on a given potatohead. Each one copies all the attributes except the new one specified and produces a potatohead with the new attribute. In all the function applications below, *aph* is a potatohead, *size* is a number, and all other parameters are symbols.

(*new-head-colour aph colour*)
(*new-head-size aph size*)
(*new-left-eye aph new-type new-col*)
(*new-l-eye-type aph new-type*)
(*new-l-eye-colour aph new-col*)
(*new-right-eye aph new-type new-col*)
(*new-r-eye-type aph new-type*)
(*new-r-eye-colour aph new-col*)
(*new-mouth aph new-type new-col*)
(*new-mouth-type aph new-type*)
(*new-mouth-colour aph new-col*)

The following function can be used to check if two potatoheads are equal:

(*ph=? ph1 ph2*)

When using `potatohead.rkt`, you cannot use *check-expect* directly with functions that produce potatoheads. To check if the result of function application (*my-ph-fun aph*) is *ph1*, you can use the following:

(*check-expect (ph=? (my-ph-fun aph) ph1) true*)

It is always possible to use *check-expect* on functions that produce numbers or symbols, as in the following:

(*check-expect (what-head-colour myph1) 'black*)

For your convenience, the following constants have been included in the teachpack: *onepotato*, *monoone*, *leftwinkone*, *rightwinkone*, *sleepone*, *twopotato*, *monotwo*, *leftwinktwo*, *rightwinktwo*, *sleeptwo*, *threepotato*, *monothree*, *leftwinkthree*, *rightwinkthree*, *sleepthree*.