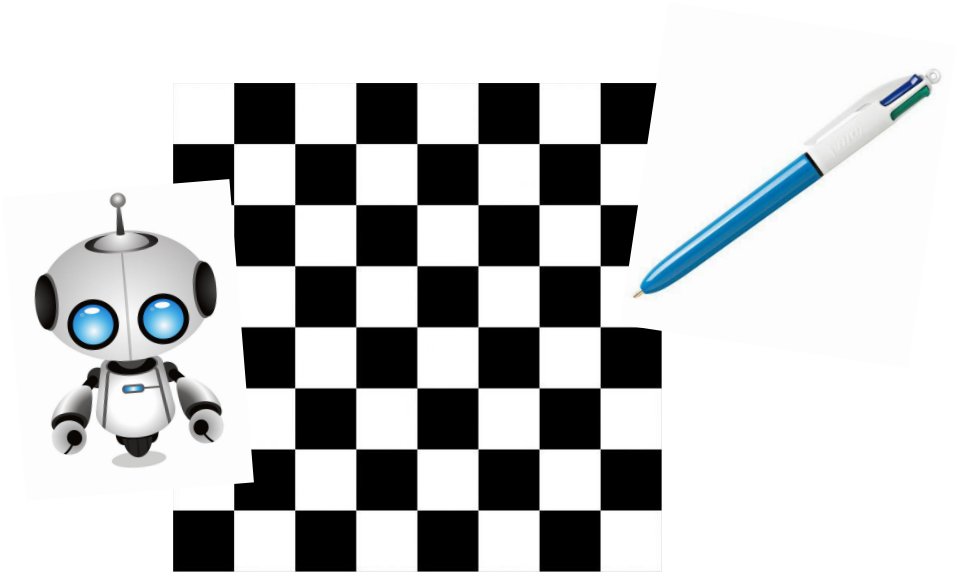
Programación

Introducción a la sintaxis estricta

Universidad Nacional de Quilmes

Introducción a QDraw



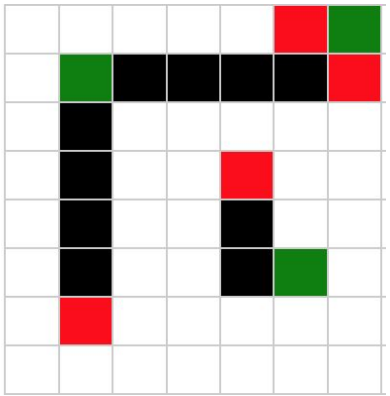
Lenguaje QDraw

Utilizaremos este lenguaje para definir programas.
Este lenguaje consta de un tablero (una hoja cuadriculada) y un autómata, representado por un cabezal.
El objetivo consiste en que el autómata se **desplace** por el tablero para ejecutar acciones sobre sus casilleros (celdas).
Las acciones que puede realizar son **pintar** y **despintar**. De esta manera se podrán realizar diferentes dibujos.

Tablero

El tablero consiste en una superficie de tamaño variable (ancho y alto) con celdas. Cada celda puede estar en blanco o pintada de algún color. Sólo maneja los colores **Negro**, **Rojo** y **Verde**.

Ejemplo:
Un tablero con un dibujo abstracto. El dibujo ocupa 7 celdas de alto (eje y) y 6 celdas de ancho (eje x).



Autómata: Cabezal

Podemos pensar el cabezal como una lapicera con puntas de múltiples colores, que se posiciona sobre una y sólo una celda del tablero para cada acción. Además de las acciones de **pintar** y **despintar** la celda, también cuenta con las acciones de **desplazamiento**. Se puede **mover hacia arriba, abajo, izquierda y derecha** para posicionarse en cada celda.

Lenguaje QDraw: instrucciones primitivas

El cabezal cuenta con un set limitado de **instrucciones primitivas**, es decir las instrucciones **más atómicas** con las que cuenta el lenguaje:

- **`MoverDerecha`**: Mueve el cabezal una celda hacia la derecha a partir de donde se encuentra ubicado
- **`MoverArriba`**: Mueve el cabezal una celda hacia arriba a partir de donde se encuentra ubicado
- **`MoverIzquierda:`** Mueve el cabezal una celda hacia la izquierda a partir de donde se encuentra ubicado
- **`MoverAbajo`**: Mueve el cabezal una celda hacia abajo a partir de donde se encuentra ubicado
- **`Limpiar`**: Despinta la celda actual sin importar el color que tenga. Es decir, elimina el color actual.
- **`PintarNegro`**: Pinta de color **negro** la celda actual
- **`PintarRojo`**: Pinta de color **rojo** la celda actual
- **`PintarVerde:`** Pinta de color **verde** la celda actual

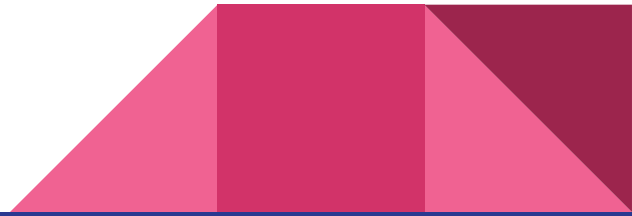
Nota: se entiende por **celda actual**, a la celda **donde se encuentra ubicado el cabezal**

Pero…¿qué es una instrucción primitiva?

Repasemos: ¿recuerdan las características de un autómata?
El autómata entiende sólo un conjunto limitado de instrucciones atómicas.
A este conjunto se lo denomina las primitivas del lenguaje.

Es como el lenguaje nativo del autómata. En este caso de nuestro cabezal.

Las primitivas que forman parte del lenguaje, tienen una sintaxis formal y estricta



Sintaxis estricta

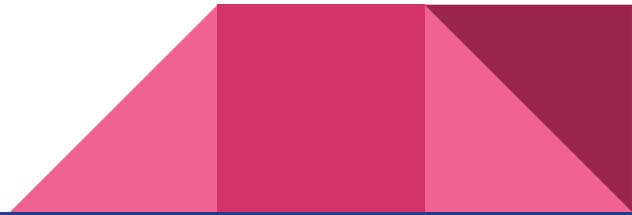
La mayoría de los lenguajes generales no usan un entorno gráfico como Lightbot, sino que se basan en una sintaxis estricta que debe utilizarse para que el autómata entienda las instrucciones que queremos darle.

QDraw es un lenguaje con sintaxis estricta.

Programa

Todo programa QDraw inicia con la palabra reservada **programa**, seguida de un bloque de código con las instrucciones a ejecutar.

Palabra reservada significa que debemos escribir esa palabra **exactamente** como lo indica su sintaxis y no se puede usar para otra cosa que no sea su definición.



Bloque de código

Un **bloque de código (fuente)** es un conjunto de instrucciones que se agrupan de manera secuencial, colocándolas entre llaves.

He aquí un ejemplo:

```
programa {
    bloque de código
}
```

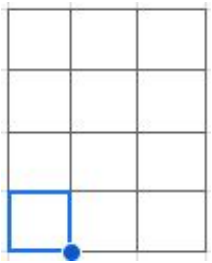
Primer programa

El propósito de este programa es dibujar 3 puntos suspensivos multicolores:

```
programa {
    PintarNegro
    MoverDerecha
    PintarRojo
    MoverDerecha
    PintarVerde
}
```

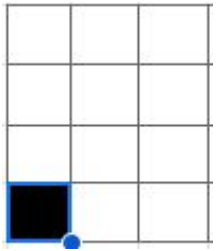
Simulación de la ejecución del programa

A continuación, se refleja de manera gráfica, cómo sería la ejecución del programa en cada instante, es decir, instrucción por instrucción:

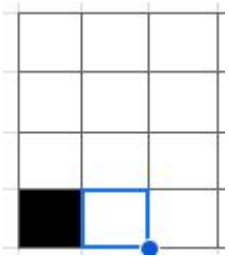


Estado inicial

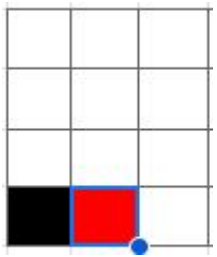
`PintarNegro`



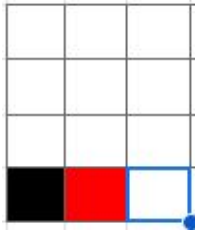
`MoverDerecha`



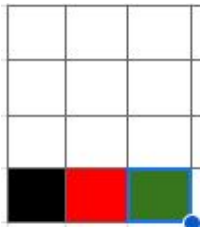
`PintarRojo`



`MoverDerecha`



`PintarVerde`



Estado final

¿Qué significa ejecutar un programa?

Un programa pasa por **2 etapas**, en el siguiente orden:

1) La definición del programa
2) La ejecución del programa

- Definir un programa, implica escribir el código fuente de su algoritmo (ya sea en papel, en un editor de texto) en un lenguaje de programación. En nuestro caso en QDraw.
- La ejecución de un programa implica que un autómata ejecute cada una de sus instrucciones. Por lo que, al finalizar la ejecución se verán reflejados los cambios en el tablero, a partir del cual podemos concluir si cumplimos o no con el propósito buscado.

En el ejemplo anterior, el estado inicial del tablero indica que el mismo está vacío, pero al finalizar la ejecución del programa, su estado final cambia, dado que contiene el dibujo.

Nota: QDraw no cuenta con una herramienta que permite ejecutar programas.
Por lo que debemos simular dicha ejecución, para probar nuestros programas.

Errores (bugs)

Errores sintácticos

La sintaxis del lenguaje es estricta. Si no respetamos la sintaxis generamos un error sintáctico y el código es inválido. De esta manera, el código ni siquiera comienza a ejecutarse, pues está mal. El autómata no puede interpretar lo que se definió.

Ejemplos:

- `{ } programa` → El orden de la palabra programa y su bloque, son incorrectos
- `prog {}` → La palabra **prog** no existe, sino **programa**
- `programa ()` → El bloque se delimita con llaves, no paréntesis
- `PintarAzul` → Esta instrucción no existe en Qdraw, no sabe interpretarla
- `MoverHaciaArriba` → Tampoco existe la instrucción. Se escribe MoverArriba

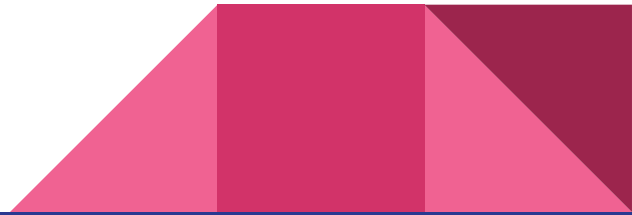
Pensando en los límites

- ¿Qué pasa si le decimos al cabezal que se mueva a una dirección que se encuentra fuera de los límites del tablero? Por ejemplo que se mueva a la derecha cuando ya no quedan celdas hacia la derecha.
- ¿Qué pasa si le decimos que despinte una celda que no está pintada de ningún color?

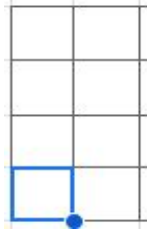
Errores lógicos

Son aquellos que se dan como resultado de ejecutar un programa sobre un tablero inválido, como cuando el cabezal intenta moverse a una celda inexistente, o intenta despintar una celda que no está pintada.

Por ejemplo, el código visto anteriormente, produciría un error lógico si se ejecutara sobre un tablero con sólo 2 celdas de ancho

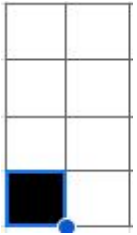


Ejecución sobre un tablero insuficiente

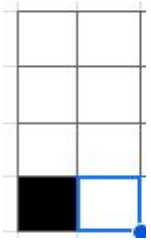


Estado inicial

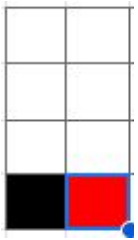
`PintarNegro`



`MoverDerecha`



`PintarRojo`



`MoverDerecha`

`PintarVerde`

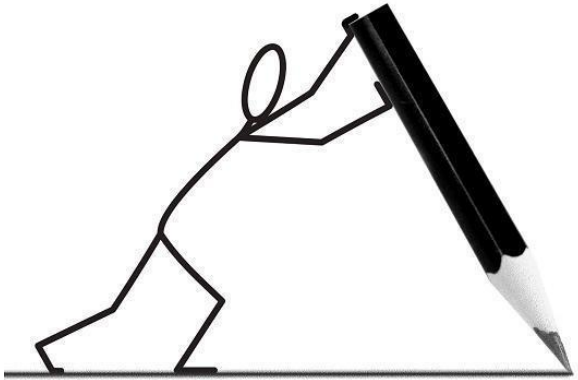
Jamás llega a ejecutarse

Errores

En el caso de Lightbot al caminar de más, no pasa nada, el robot queda en el lugar.

Pero en el caso de QDraw, si se realiza alguna acción inválida, el programa falla, el cabezal explota por los aires y el/la programador/a muere en el acto... bueno, no tan así 🤭, pero **no podemos tener instrucciones inválidas**.

Documentación



Comentarios

Una manera de **documentar** es mediante **comentarios**. Los comentarios consisten en un texto que el/la programador/a puede agregar a su código, y son útiles para comunicar información sobre el programa.

¿A quiénes comunicamos?

La documentación no le sirve al autómata, sino a otras personas, o a uno/a misma, y comunica sobre las intenciones del programa.

La **documentación** forma parte de un programa y es preciso que **sea clara y precisa.**

Nota: se puede escriben en español, inglés, etc.

Comentarios: Sintaxis

Los comentarios en QDraw están delimitados por los siguientes caracteres:

`/*` *(Símbolo de apertura - inicio)*

`*/` *(Símbolo de cierre)*

Todo lo que está entre esos dos símbolos es ignorado por el cabezal (autómata).

Ejemplo:

```
programa {
    /* Acá va la documentación del programa */
        PintarNegro
        ...
    }
```

Comentarios: Sintaxis

Los comentarios pueden ocupar múltiples líneas, y puede haber más de uno en un mismo programa.

```
programa {
    /* Esto es un comentario y será ignorado por el autómata.
       Puede ocupar más de una línea. */
    PintarNegro
    /* Acá podemos incluir otro comentario. */
    MoverDerecha
    ...
}
```

Propósito

Propósito: definición

Como ya vimos, el resultado de ejecutar un programa implica un cambio de estado en el tablero. Por lo cual, el **propósito** debe indicar todos los cambios que se dan en el mismo. De esta manera se deberá detallar:

1. El **objetivo final** del programa, es decir lo que realiza, independientemente de cómo lo realiza. Comunica el **qué** y no el **cómo**.
2. El **estado final** en el cual quedará el cabezal sobre el tablero al finalizar la ejecución del programa, pero **sólo en caso que difiera de su estado inicial**. En caso contrario, no habría un cambio de estado por lo que no habrá nada que comunicar, **quedando sin estado final**

Precondición

Precondición: definición

La **precondición** es parte de la documentación del programa, y es aquello que el programa **necesita / requiere sí o sí para cumplir con su propósito**.
Son las condiciones con las que debemos contar, los requisitos necesarios, para pasar del estado inicial al estado final, luego de ejecutar el programa.
Esto implica que si no tenemos en cuenta estas condiciones, el programa puede fallar o realizar otra cosa. **Es aquello que se necesita para que el programa termine de forma exitosa**, evitando que haga "BOOM".
Como la precondición es parte de la documentación, y como tal, de la información que necesitamos comunicar, será necesario mencionar, además, el **estado inicial del cabezal**, a partir del cual el programa requiere las condiciones mencionadas previamente.
Esta información se deberá agregar siempre.

Documentación del programa de ejemplo

```
programa {
/*PROPÓSITO: Dibujar 3 puntos suspensivos multicolores. El cabezal
finaliza en el 3er punto.
PRECONDICIÓN: El cabezal inicia en el 1er punto, y debe haber al
menos 2 celdas hacia su derecha.*/
    PintarNegro
    MoverDerecha
    PintarRojo
    MoverDerecha
    PintarVerde
}
```

Tener en cuenta que cuando dice: “su” derecha, hace mención a la posición inicial del cabezal.

Como el cabezal inicia en una celda diferente a donde termina, se informa dicho cambio en el propósito.

Documentación con otro ejemplo

Ahora veamos una variación en el programa:

```
programa {
/*PROPÓSITO: Dibujar 3 puntos suspensivos multicolores. El cabezal finaliza en el 3er punto.
PRECONDICIÓN: El cabezal inicia en el 1er punto, y debe haber al menos 2 celdas hacia
su derecha.*/
      PintarNegro
      MoverDerecha
      PintarRojo
      MoverDerecha
      PintarVerde
      MoverIzquierda
      MoverIzquierda
}
```

Se agregaron estas 2 instrucciones, volviendo a la celda inicial.
Es decir que el cabezal inicia y termina en la misma celda. No hay un cambio de estado en el cabezal luego de la ejecución del programa. Sólo en el tablero, producto del dibujo.

Ante este cambio, la precondición no cambia, pero sí el propósito. Ya no es necesario informar el estado final del cabezal, dado que éste no varía con respecto a su estado inicial. De esta manera, se asume que finaliza en el mismo lugar donde inició. Evitando así redundancias en la comunicación, e informando sólo el cambio producido en el tablero: El dibujo.

Ejemplos varios

- */*__PROPÓSITO__: Dibujar un cuadrado de 3x3.*
  ***PRECONDICIÓN**: El cabezal inicia en la esquina inferior izquierda del cuadrado, y debe haber al menos 3 celdas hacia arriba y 3 celdas hacia la derecha desde su inicio.*/*

- */*__PROPÓSITO__: Dibujar un perro negro. El cabezal finaliza en la pata del perro*
  ***PRECONDICIÓN**: El cabezal inicia en el hocico del perro, y debe haber al menos 4 celdas hacia abajo, 1 celda hacia arriba y 3 celdas hacia la derecha desde su inicio.*/*

- */*__PROPÓSITO__: Polinizar todas las flores del jardín. Susy finaliza en la llegada.*
  ***PRECONDICIÓN**: Susy inicia en la entrada del jardín, y debe haber al menos 10 parcelas hacia la izquierda, 3 hacia abajo y 10 hacia la derecha desde su inicio.*/*

- */*__PROPÓSITO__: Comer todas las estrella del mar. José finaliza en la 4ta estrella de la 4ta fila del mar.*
  ***PRECONDICIÓN:** José inicia en el mar lejos de las estrellas, y el mar debe tener 4 filas de 4 estrellas cada una .*/*

En algunos casos, el cabezal está representado por un sujeto dado en el contexto del problema (enunciado). Como es el caso de Susy o José.

A modo de resumen

- Todo programa se conforma de su código fuente más **la documentación**. No es opcional.
- En la documentación se describe el **propósito y la precondición**.
- En el propósito se indican todos los cambios producidos luego de ejecutar el programa. Es decir, el dibujo o resultado obtenido sobre el tablero, y en caso de suceder, el estado final del cabezal, **sólo en caso que éste varíe del inicial**.
- No interesa qué pasos realiza el cabezal para lograr el estado final. **Sólo su objetivo. No describir el algoritmo.**
- No interesa qué condiciones se necesitan para que el programa finalice de forma exitosa.
- Las precondiciones son necesarias cuando una instrucción es parcial, es decir cuando un programa puede fallar. Son las condiciones necesarias para evitar que el programa haga "BOOM". Ejemplo, en los **algoritmos donde hay desplazamiento**.
- **No siempre hay precondiciones**. Es documentación opcional.

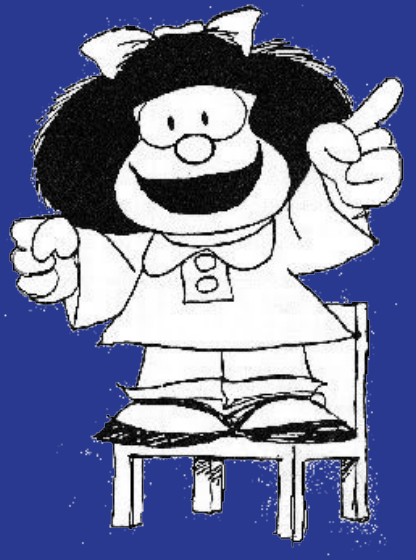
Basta de teoría, ahora a ejercitar

Ejercicios para precalentar

- Escribir un programa que dibuje una **linea vertical** de 3 de alto de color **rojo**.
- Escribir un programa que dibuje una **línea horizontal** de 7 celdas de largo de color **verde**

Para reflexionar...

"Si vas a cometer errores que sean nuevos"



Programación

Introducción a la sintaxis estricta

Universidad Nacional de Quilmes