IIP (E.T.S. d'Enginyeria Informàtica)
Curs 2019-2020

Pràctica 2. Objectes, classes i programes. L'entorn BlueJ

Professors d'IIP
Departament de Sistemes Informàtics i Computació
Universitat Politècnica de València

Índex

1 Context i treball previ a la sessió de pràctiques

En el marc acadèmic, aquesta pràctica és la segona del curs i té com a objectiu principal la definició i ús de classes utilitzant l'entorn de programació *BlueJ*. Amb aquesta pràctica es pretén que l'alumne es familiaritze amb els aspectes bàsics de *BlueJ*, i siga capaç d'utilitzar-lo d'ara endavant com a entorn de treball per a les pràctiques de l'assignatura.

L'alumne serà capaç d'utilitzar l'entorn per:

- crear un projecte *BlueJ* que incloga un paquet amb les classes que es proporcionen a l'inici de la pràctica,
- editar alguna de les classes existents,

- crear objectes i executar mètodes sobre ells utilitzant el banc d'objectes de *BlueJ* (*Object Bench*),
- avaluar expressions utilitzant l'intèrpret d'instruccions de *BlueJ* (*Code Pad*),
- generar de forma automàtica la documentació del projecte,
- validar el funcionament d'una classe o de qualsevol dels seus mètodes, utilitzant el depurador de *BlueJ* (*Debugger*).

Per tal d'aprofitar al màxim la sessió de pràctiques, s'aconsella a l'alumne que, abans, faça una lectura comprensiva d'aquest butlletí.

2 Desenvolupament d'un projecte *BlueJ*

Com ja s'ha indicat en la secció 7 del butlletí de la pràctica 1 (*Introducció: Linux, Java i BlueJ*), un projecte *BlueJ* consta d'un conjunt de classes relacionades entre sí i l'entorn *BlueJ* no permet només desenvolupar el projecte esmentat (crear, compilar, executar, depurar, documentar, etc. el projecte i les seues classes) sino que també *permet interactuar amb qualsevol mètode (atribut u objecte) de qualsevol de les classes incloses en el projecte.* A més, en un projecte *BlueJ* les diferents classes que el composen poden estar organitzades en *paquets* o *llibreries* de classes, seguint l'organització de classes de Java, permetent així la seua posterior reutilització des d'altres classes.

L'objectiu d'aquesta pràctica és que l'alumne es familiaritze amb tots aquestos conceptes fent ús de l'entorn *BlueJ*.

2.1 Invocació a *BlueJ* i creació d'un projecte

Com ja es va indicar en la pràctica anterior, *BlueJ* es pot invocar des del menú desplegable de l'entorn gràfic del sistema `Aplicaciones - Programación - BlueJ 4.2.1` o bé des de la línia d'ordres com segueix:

```
bluej &  ó  bluej nomProjecte &
```

Noteu que la segona opció és equivalent a invocar a *BlueJ* sense arguments i després, utilitzant l'opció `Projecte - Obre Projecte...` del menú, seleccionar el projecte `nomProjecte`. Si es vol obrir amb *BlueJ* una aplicació ja existent però que **no** ha estat desenvolupada amb *BlueJ*, s'ha d'invocar a *BlueJ* sense arguments i aleshores obrir l'aplicació existent amb l'opció `Projecte - Obre No BlueJ...` del menú.

Nota: Es pot canviar l'idioma de l'entorn *BlueJ* fent servir l'opció `Eines - Preferències - Interface` o en anglès: `Tools - Preferences - Interface`.

Activitat #1

1. Descarregar el fitxer `iip.jar` disponible a la carpeta `Recursos/Laboratorio/ Práctica 2/Valencià/Codi` de PoliformaT en el teu directori `$HOME/DiscoW`.
2. Invocar a *BlueJ* sense arguments.
3. Obrir el fitxer `iip.jar` amb l'opció `Open ZIP/JAR...` del menú.

`iip` és un projecte *BlueJ* amb un paquet anomenat `pract2`. En la finestra principal de *BlueJ* apareixen les icones de cadascuna de les classes del paquet (Figura 1) i una icona d'una carpeta amb el text `<go up>` que permet accedir al projecte `iip`. Noteu que les icones

associades als fitxers `.java` apareixen ratllades si encara no han estat compilats. Les fletxes indiquen les relacions d'ús que existeixen entre les classes. Si d'un fitxer només es troba l'arxiu `.class`, llavors és que no es troba el codi font del mateix (no pot editar-se, però sí pot utilitzar-se) i apareix marcat amb el text `(no source)`.

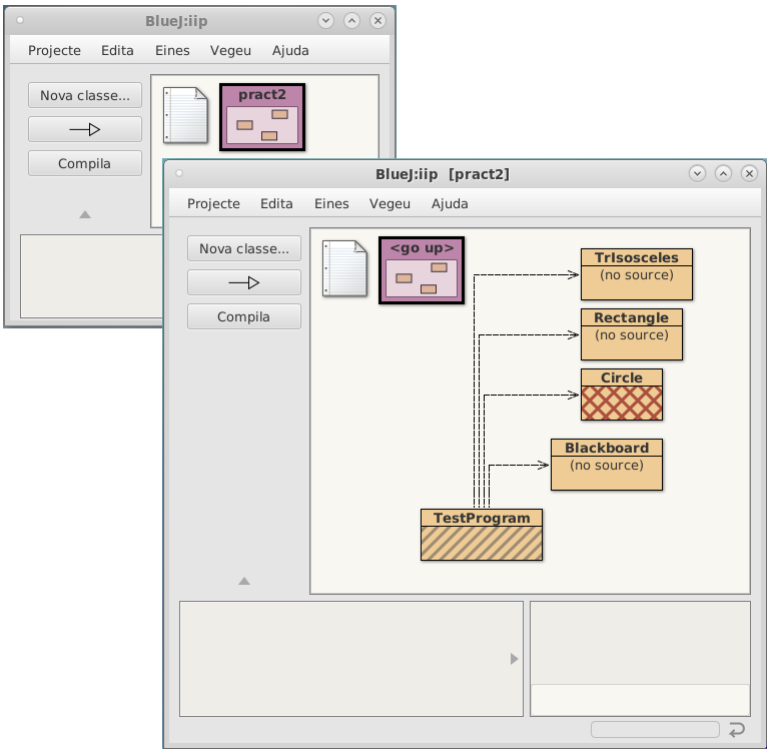


Figura 1: Projecte `iip` que inclou el paquet `pract2`.

2.2 Operacions d'edició i compilació

Per editar una classe, en el nostre cas `Circle`, es pot usar l'opció `Obre Editor` del menú de la classe, o també fer doble clic sobre la icona de la classe.

Una classe es pot compilar des del propi editor o, *si es desitja compilar tot el projecte*, des de la barra d'eines de la finestra principal de *BlueJ*. *BlueJ* realitza una precompilació del codi i, si es produeixen errors sintàctics, el propi editor marca aquests errors amb un senyal roig a l'esquerra del text i un xicotet subratllat on considera que hi ha un error. Per exemple, si el compilador detecta que falta un punt i coma, en situar el cursor en la part subratllada apareix el missatge en un xicotet requadre com pot veure's en la Figura 2. És més, si es clica en la paraula `Errors`, en la part inferior esquerra de la zona d'informació de l'editor, es pot anar passant d'error en error detectat. Una vegada corregits aquests errors, es pot fer clic al botó de compilar en la part superior de la finestra i, llavors, o bé pot resultar el codi correctament compilat o poden aparèixer nous errors.

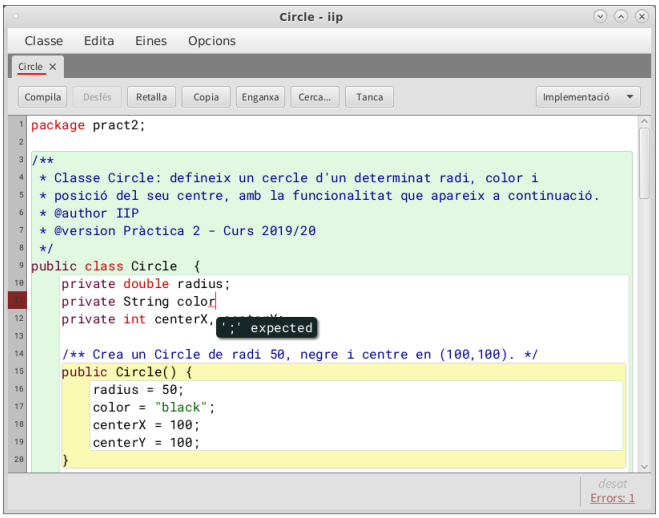


Figura 2: Compilació de la classe `Circle`.

Activitat #2

1. Comprovar que la primera línia de les classes `Circle` i `TestProgram` inclou la directiva del compilador per tal d'indicar que són classes pertanyents a un paquet:

   ```
   package pract2;
   ```

2. Compilar la classe `Circle` i corregir els possibles errors de compilació.
3. Compilar la classe `TestProgram`. Observar què passa a la part central de la finestra principal de *BlueJ*.

Una de les característiques que ofereix l'editor de *BlueJ* és l'autocompletat de codi que s'activa en polsar les tecles `Ctrl-Space`.

Activitat #3

Encara que la classe `TestProgram` no té cap error de compilació, sí que té un error lògic: la intenció del programador era mostrar per pantalla el perímetre del cercle creat, però realment no és així. En aquesta activitat es corregirà aquest error.

1. Editar la classe `TestProgram` i completar la instrucció que mostra per pantalla el perímetre de l'objecte `Cercle c`. Escriure `c.` i polsar `Ctrl-Space`. Comprovar que, com en la Figura 3, apareix un llistat de tots els mètodes disponibles en la classe junt amb una xicoteta descripció dels mateixos. Seleccionar el que interesse i després de polsar `Enter`, *BlueJ* ho autocompletarà.

2. A continuació, des de l'opció `Eines - Checkstyle` en la vista de projecte, comprovar si el codi d'alguna classe té errors d'estil i corregir-los si és el cas (Figura 4).

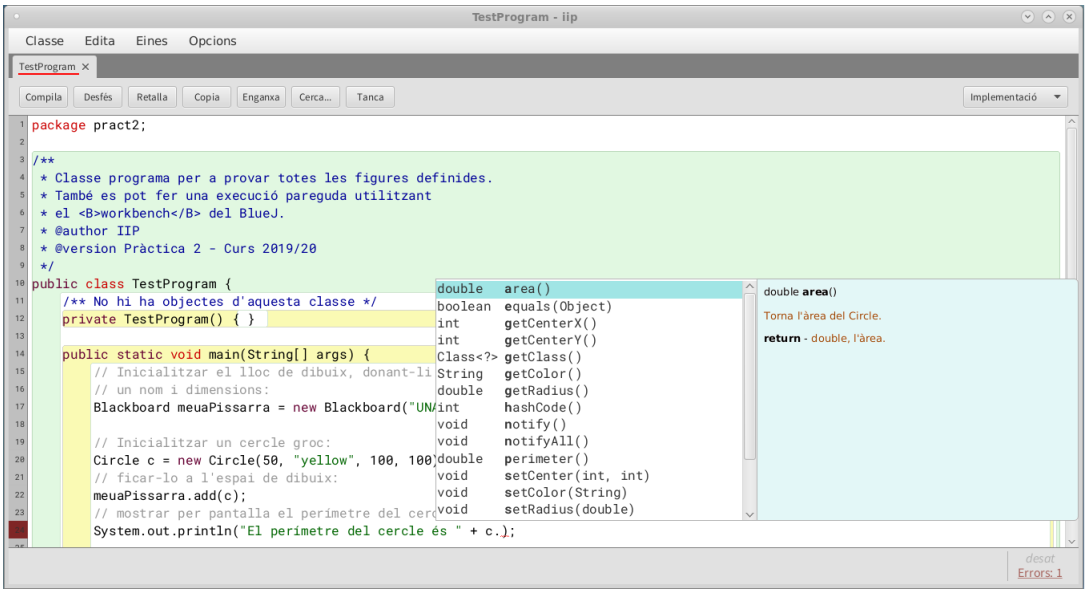


Figura 3: Autocompletat de codi en l'editor de *BlueJ*.

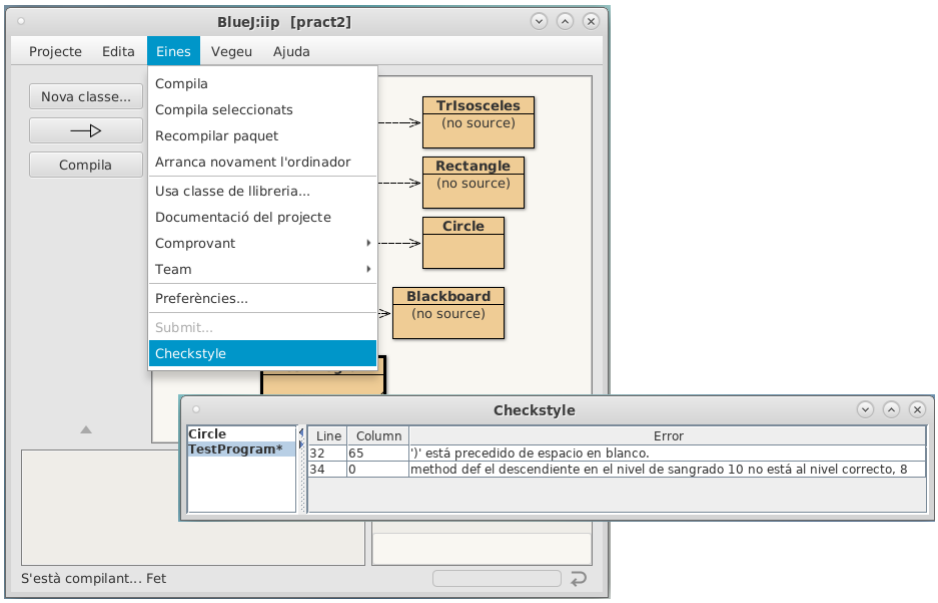


Figura 4: Comprovació de l'estil del codi en *BlueJ*.

2.3 Execució d'una classe: crides a `main`

Com ja s'ha dit anteriorment, al marcar la icona d'una classe i fer clic amb el botó dret del ratolí, s'obté el menú de la classe; doncs bé, de la llista d'operacions que aquest conté destacarem ara l'operació de crida al mètode `main` de la classe; *es pot executar una classe des del seu menú*. No és necessari passar cap argument al `main` de `TestProgram`, només apareixen les claus.

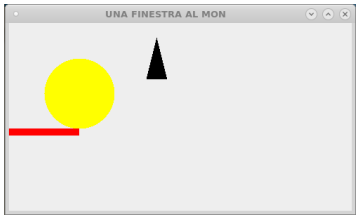
A la Figura 5(a) es mostra el menú emergent de la classe `TestProgram` i a la figura 5(b) la finestra de la crida al mètode `main` de la classe `TestProgram`.

Activitat #4

Executar el mètode `main` de la classe `TestProgram`. El resultat ha de ser com el que apareix a la Figura 5(c) i en el terminal s'ha de mostrar el missatge de la Figura 6.

(a) Menú de la classe.

(b) Crida al mètode `main`.



(c) Resultat de l'execució.

Figura 5: Execució de la classe `TestProgram`.

En cas de que el programa sol·licite dades des del teclat o mostre per pantalla algun resultat (Figura 6), apareix de forma automàtica un terminal de text. Si no apareix, s'ha de seleccionar l'opció `Mostrar Terminal` del menú `Vegeu`. És important assenyalar que en la versió 4.1.2 de *BlueJ*, quan cal introduir alguna dada des de teclat, es fa en una línia addicional en la part inferior de la finestra de terminal que, si el programa no necessita entrada de dades en aquest moment, apareix atenuada.

Figura 6: Finestra de terminal de *BlueJ*.

Si s'executa novament el `main` de `TestProgram`, es pot comprovar que en la finestra de terminal es torna a escriure el missatge. Per tal que, en cada nova execució, s'esborre del terminal el resultat de l'execució anterior, seleccionar, des del menú `Opcions` de la finestra de terminal, l'opció `Neteja la pantalla en cridar el mètode`, com en la Figura 7. Per a futures pràctiques, seleccionar també l'opció `Buffering il·limitat`.

Figura 7: Opcions de la finestra de terminal de *BlueJ*.

2.4 L'opció *Eines*. Generar documentació

De les diferents utilitats de l'opció `Eines` cal destacar `Documentació del projecte`, que genera el subdirectori `doc` amb la documentació sobre les classes en format `html`. Noteu que la documentació individual d'una classe també es pot generar a l'editar la classe i seleccionar, en lloc d'`Implementació`, l'opció `Interfície` o també seleccionant `Eines - Canvia a vista d'interfície`.

Activitat #5

Generar la documentació del projecte i consultar-la. El resultat serà com el que apareix a la Figura 8. Fent "click", per exemple, en l'enllaç de la classe `Circle` es pot consultar la seua documentació.

Figura 8: Documentació del projecte `pract2`.

2.5 L'opció *Ajuda*

La utilitat `Llibreries de classes Java` permet accedir a la documentació de Java. En la instal·lació per defecte aquesta ajuda es troba a `https://docs.oracle.com/en/java/javase/11/docs/api/`. Si es vol canviar a un directori local, es por fer des de l'opció `Eines/Preferències/Miscel·lània` del menú.

Altres utilitats d'`Ajuda` permeten consultar el manual de *BlueJ* així com accedir a la seua web `www.bluej.org`.

3 Ús del banc d'objectes (*Object Bench*)

Una de les característiques més interessants de l'entorn *BlueJ* és que permet interactuar amb objectes aïllats de qualsevol classe i executar els mètodes que sobre ells s'hagen definit; d'aquesta manera es pot comprovar la funcionalitat de la classe abans d'escriure qualsevol aplicació que la utilitze.

3.1 Operacions d'una classe

Per accedir a les operacions aplicables a una determinada classe hi ha que marcar la icona de la classe i fer clic amb el botó dret del ratolí. Apareix una llista amb les operacions constructores de la classe i altres operacions permeses per l'entorn com, per exemple, esborrar la classe o compilar-la (Figura 9).

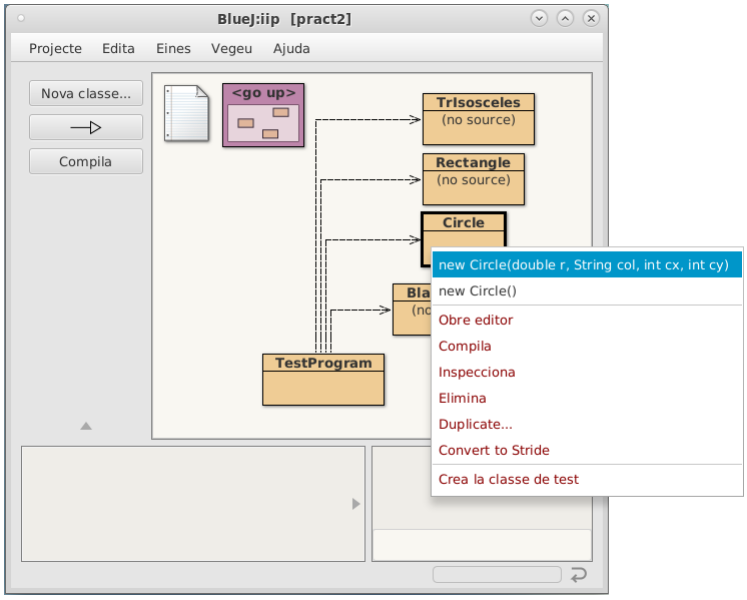


Figura 9: Menú de la classe `Circle`.

3.2 Creació d'un objecte

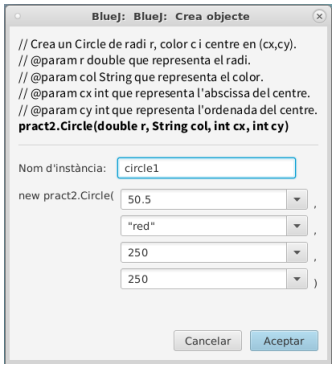
Si es vol crear un objecte s'ha de seleccionar d'aquest menú una de les operacions constructores i seguir el quadre de diàleg que s'obri (Figura 10(a)). En concret, es demana un nom per a l'objecte. Quan es crea aquest objecte, apareix a la part inferior esquerra de la pantalla principal de *BlueJ*, a la zona coneguda com *banc d'objectes* (*Object Bench*) (Figura 10(b)).

3.3 Execució de mètodes sobre l'objecte

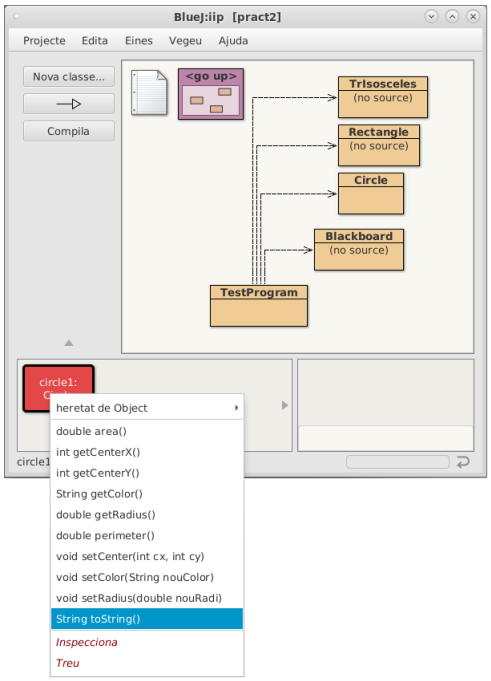
Si es clica amb el botó dret del ratolí sobre l'objecte creat s'accedeix als mètodes que es poden executar sobre el mateix (Figura 10(b)). Per executar un d'ells només cal seleccionar-lo. Si l'objecte hereta mètodes d'altres classes també apareixen a través de submenús.

3.4 Observació de l'estat de l'objecte

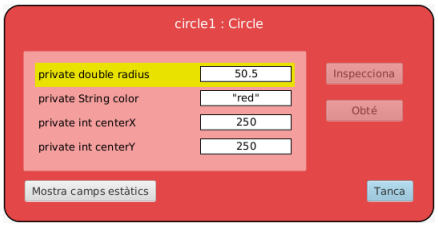
Per depurar els mètodes dissenyats es pot utilitzar l'opció `Inspecciona`. Aquesta operació permet conèixer els valors dels camps (atributs) dels objectes (Figura 10(c)).



(a) Creació d'un objecte `Circle`.



(b) Banc d'objectes i menú de l'objecte `Circle`.



(c) Inspecció de l'objecte `Circle`.

Figura 10: Creació d'un objecte en el banc d'objectes de *BlueJ*, menú i observació de l'estat d'aquest objecte.

Activitat #6

1. Crear un objecte de la classe `Circle` de radi 50.5, color roig i amb centre en $(250, 250)$.
2. Consultar els valors dels atributs de l'objecte de tipus `Circle` creat.
3. Executar el mètode `toString()` definit a la classe `Circle` sobre l'objecte creat.
4. Modificar el radi del `Circle` perquè valga 30.0.
5. Executar novament el mètode `toString()`.
6. Crear un objecte de la classe `Blackboard` amb títol "Dibuix" i dimensió 500 x 500. Per tal que puga veure's l'efecte de l'ítem 8, l'objecte creat no ha de tancar-se.
7. Consultar els valors dels atributs de l'objecte creat.
8. Afegir l'objecte `Circle` al `Blackboard`.

4 Ús de l'avaluador d'expressions (*Code Pad*)

La *zona de codi* (*Code Pad*) de *BlueJ* està situada al cantó inferior dret junt al banc d'objectes (Figura 11). Si no es mostra, s'ha de seleccionar l'opció `Mostra el quadern de notes` del menú `Vegeu`.

En la línia especial situada en la part inferior d'aquesta zona es pot introduir tant una expressió com una instrucció en Java, on poden aparèixer objectes del banc d'objectes; polsant *Enter*, cada línia serà avaluada i es mostrarà el valor resultant, seguit pel seu tipus (entre parèntesi), o un missatge d'error si l'expressió/instrucció és incorrecta.

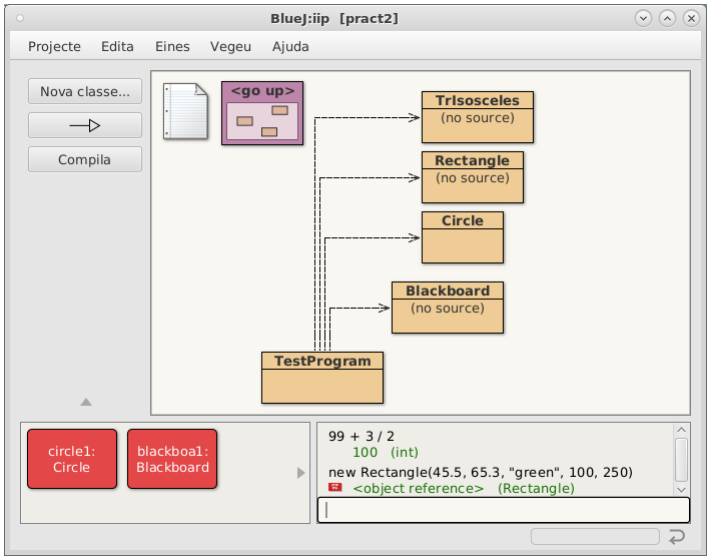


Figura 11: Zona de codi de *BlueJ*.

Alguns resultats d'expressions són objectes en lloc de valors simples. En aquest cas l'objecte es mostra com una referència a objecte `<object reference>`, seguida pel tipus de l'objecte i es mostra una icona menuda representativa de l'objecte al costat de la línia de resultat. Aquesta icona es pot utilitzar ara per continuar treballant amb l'objecte resultant. Es pot arrossegar la icona al banc d'objectes. Això situarà l'objecte al banc, on estarà disponible per a futures crides als seus mètodes, bé via el seu menú emergent o bé via la zona de codi.

Activitat #7

1. Quin resultat s'obté en avaluar cadascuna de les expressions següents a la zona de codi de *BlueJ*?

1	`8 % 3`	6	`9 / 2`
2	`(int) 98.67`	7	`9.0 / 2.0`
3	`Math.round(98.67)`	8	`9 / 2.0`
4	`Math.sqrt(121)`	9	`9 / (double) 2`
5	`Math.sqrt(-5)`	10	`9 / 0`

2. Definir les variables enteres `x` i `y`, amb valors 4 i 6, respectivament, i escriure una expressió aritmètica per tal de calcular l'expressió algebraica següent:

$$\frac{x^2 - y}{x} \tag{1}$$

3. Definir les variables enteres `a`, `b` i `c`, amb valors 2, `-7` i 3, respectivament, i escriure una expressió aritmètica per tal de calcular l'expressió algebraica següent:

$$\frac{-b + \sqrt{b^2 - 4ac}}{2a} \tag{2}$$

4. Escriure en la zona de codi una instrucció que mostre per pantalla (finestra de terminal de *BlueJ*) el radi de `circle1`, el `Circle` situat en el banc d'objectes des de l'Activitat #6.
5. Executar en la zona de codi el mètode `toString()` sobre `circle1` i arrossegar la icona de l'objecte `String` resultat al banc d'objectes.
6. Crear un nou objecte `Circle` a la zona de codi i arrossegar la seua icona al banc d'objectes. Anomenar-lo `circle2`.
7. Escriure a la zona de codi una expressió que torne el color de `circle2`.
8. Executar en la zona de codi el mètode `toString()` sobre `circle2`. La icona de l'objecte `String` resultat es pot també arrossegar al banc d'objectes.

5 Ús del depurador (*Debugger*)

El *depurador* de *BlueJ* és una eina senzilla, però de gran utilitat a l'hora de validar el funcionament d'una classe (és a dir, el del seu `main`) o qualsevol mètode d'aquesta. El seu ús permet, bàsicament,

- observar l'execució de qualsevol mètode, és a dir, *fer la seua traça per uns valors donats*, bé pas a pas, bé de només alguna(es) de les seues línies;
- inspeccionar la seqüència de crides associada a la invocació del mètode en execució;
- comprovar el pas de paràmetres i els valors que prenen les variables locals al mètode en execució.

Per aconseguir aquests resultats, el depurador disposa de les següents funcions:

1. **Establir punts de ruptura**. Només quan es deté l'execució d'un mètode en un cert punt del seu codi és possible observar l'estat de la seua execució en aquest punt. El depurador proporciona una funció que deté l'execució d'un mètode en un cert punt del codi, o equivalentment, *estableix punts de ruptura*.

   A *BlueJ* els punts de ruptura s'estableixen en l'anomenada *àrea de punts de ruptura* de l'editor, situada a l'esquerra del text; només cal fer-hi clic, a l'altura de la línia de codi on es vol detindre l'execució d'un mètode, i apareixerà un xicotet signe d'stop com a marca de punt de ruptura. Quan durant l'execució d'un mètode s'arriba a la línia així marcada, l'execució s'interromp. A més, apareixen, una darrere l'altra,

   (a) *la finestra de l'editor*, on figura ressaltada la línia següent a la que conté el punt de ruptura, ja que és *la següent línia a executar*;

   (b) *la finestra del depurador*; els diferents tipus d'informació i botons que conté aquesta finestra es presenten a continuació.

2. **Execució pas a pas**. Una vegada detinguda l'execució, aquesta es pot reprendre pas a pas, instrucció a instrucció, el que permet seguir el codi observant com progressa l'execució, és a dir, *fer una traça* del codi.

   Per realitzar una execució pas a pas en *BlueJ* n'hi ha prou amb fer clic repetidament sobre el botó `Pas` de la finestra del depurador. Cada clic suposa l'execució d'una única línia de codi; després d'això, l'execució es torna a detindre.

   Si es vol sortir d'aquest procés, tornant a l'execució normal del mètode, només cal esborrar la marca de ruptura establerta, simplement fent clic sobre ella, i després polsar el botó `Continua` de la finestra del depurador.

3. **Inspecció de variables i del pas de paràmetres**.

   Només amb observar la finestra de depuració de *BlueJ* es pot veure la seqüència de crides associada a la invocació del mètode en execució, comprovar el pas de paràmetres i inspeccionar els valors que prenen les seues variables locals.

   Per inspeccionar qualsevol variable o paràmetre d'aquest tipus només cal fer un doble clic sobre ell, en la finestra del depurador.

Activitat #8

1. En el mètode `main` de la classe `TestProgram`, establir un punt de ruptura en les línies on es creen els objectes de tipus `Circle`, `Rectangle` i `TrIsosceles`.

2. Executar el mètode `main`. Observeu què passa quan, una vegada arribat al punt de ruptura es fa clic sobre el botó `Pas` de la finestra del depurador.

3. Per inspeccionar les variables del `main`, fer doble clic sobre elles en la finestra del depurador.

A la Figura 12 es mostra el depurador de *BlueJ* una vegada alcançat l'últim punt de ruptura en l'execució de la classe `TestProgram`.

Activitat #9

Escriure una classe programa, similar a la classe `TestProgram`, que mostre una figura formada per cercles, rectangles i triangules.

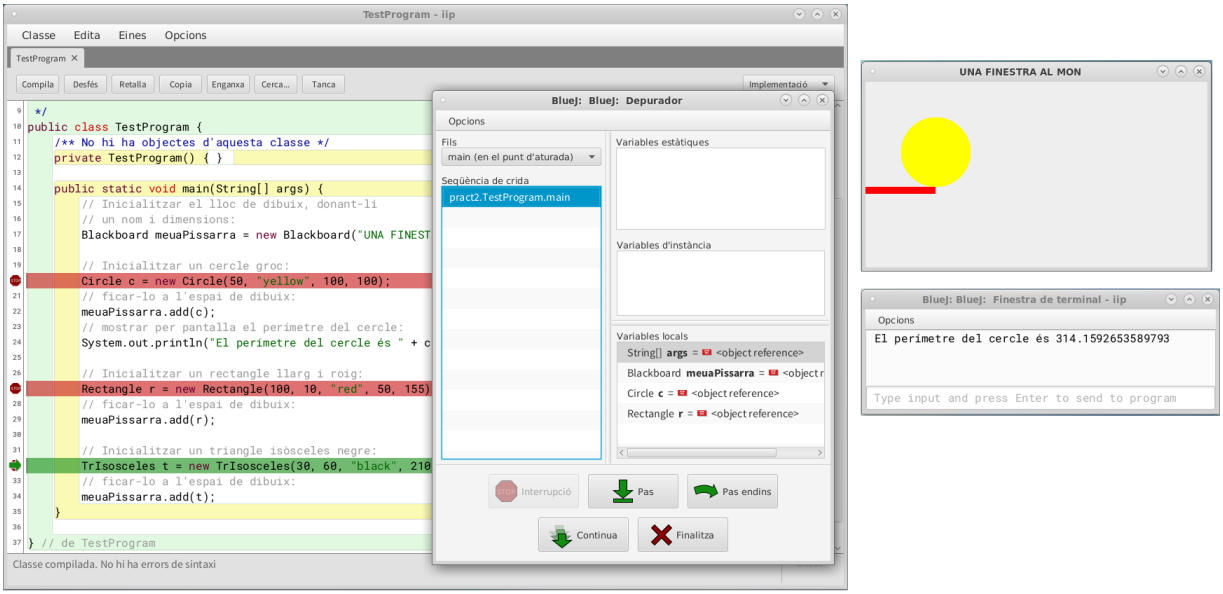


Figura 12: Depurador de *BlueJ*.