IIP (E.T.S. d'Enginyeria Informàtica)
Curs 2021-2022

Pràctica 2. Objectes, classes i programes en l'entorn BlueJ

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

Índex

1 Objectiu i treball previ a la sessió de pràctiques

L'objectiu d'aquesta pràctica és que conegues i faces servir les eines que t'ofereix *BlueJ* per a desenvolupar aplicacions Java i, així, sigues capaç d'usar-lo sense més problemes com a entorn de treball. En concret, al concloure aquesta pràctica has de ser capaç d'utilitzar *BlueJ* per tal de:

- Ampliar el teu projecte `iip` creant un nou paquet i ubicar en ell les classes d'una (xicoteta) aplicació gràfica que se't proporcionen a l'inici de la pràctica.
- Actualitzar alguna de les classes que conté el nou paquet.
- Generar de forma automàtica la documentació d'un projecte o d'una de les seues classes.
- Crear objectes i executar mètodes sobre ells utilitzant el banc d'objectes, u *Object Bench*.
- Avaluar expressions i executar instruccions Java usant la "zona de codi", o *Code Pad*, que és l'intèrpret d'instruccions de l'entorn.
- Validar el funcionament d'una classe o de qualsevol dels seus mètodes, utilitzant el depurador de *BlueJ* (*Debugger*).

Per realitzar aquesta pràctica cal que, al menys, tingues ja al teu ordinador (a la carpeta `labIIP`) el projecte *BlueJ* `iip` i que sàpigues com crear un nou paquet dins d'ell; recorda que tot això se't va explicar detalladament al realitzar la pràctica 1. A més, seria més que

convenient que llegires el butlletí d'aquesta pràctica abans de realitzar-la, ja que així podràs aprofitar millor el temps de la sessió amb el teu professor.

2 Ampliació d'un projecte *BlueJ*

Com saps, l'entorn *BlueJ* permet desenvolupar una aplicació Java com un projecte, un directori que conté tots els fitxers (`.java` i `.class`) associats a les classes Java que componen l'aplicació. En concret, *BlueJ* proporciona les eines necessàries per a:

- Organitzar les classes del projecte en *paquets* o *llibreries* de classes, a l'igual que en l'estàndard de Java, permetent així la seua posterior reutilització des d'altres classes.
- Desenvolupar el projecte, i.e. crear, compilar, executar i documentar les classes que el componen.
- Interactuar amb qualsevol element (atribut o mètode) o instància (objecte) de qualsevol de les classes de el projecte.

Després de realitzar la pràctica 1, ja tens en una carpeta (`labIIP`) del teu ordinador personal el projecte *BlueJ* `iip` de les pràctiques de l'assignatura. Per ampliar-lo, i.e. per integrar en ell de forma adequada les classes corresponents a aquesta pràctica, hauràs de començar realitzant les activitats bàsiques que s'enuncien en aquest apartat, totes elles directament relacionades amb els dos primers ítems que s'acaben de descriure.

Invocació a *BlueJ* i creació d'un (nou) paquet

El primer pas per ampliar el teu projecte *BlueJ* `iip` és actualitzar la seua estructura de paquets i classes tal com se t'indica a continuació, destacant en negreta les opcions de l'entorn *BlueJ* que no vas usar en les activitats de la pràctica 1.

Activitat #1

a) Invocar a *BlueJ* sense arguments, des del menú desplegable de l'entorn gràfic del sistema o usant la icona d'accés directe si es té.

   Nota: recorda que si el menú de la finestra principal de *BlueJ* apareix en anglès, pots canviar l'idioma amb l'opció `Tools - Preferences - Interface - Language selection`, seleccionar `Catalan` (o `Spanish`) i, després, reiniciar *BlueJ* per tal que s'aplique el canvi.

b) Obrir el teu projecte `iip` **seleccionant l'opció** `Projecte - Obre Projecte...` **del menú de la finestra principal de *BlueJ***. Verifica que estàs en el projecte `iip` i no en el paquet `pract1`, fent doble clic en la icona `<go up>` si és necessari.

c) En el projecte `iip`, crear un nou paquet de nom `pract2`; aquest nou paquet contindrà les classes amb les que es treballarà en aquesta pràctica, totes elles disponibles en PoliformaT.

d) Eixir de *BlueJ* **seleccionant l'opció** `Projecte - Surt` **del menú de la finestra principal de *BlueJ***.

e) Descarregar en el directori corresponent al paquet *BlueJ* `pract2` els fitxers disponibles en la carpeta `Recursos/Laboratorio/Práctica 2/Valencià/Codi` de PoliformaT.

f) Invocar de nou a *BlueJ* i obrir el projecte `iip`. Si tot ha anat bé, en la finestra principal de *BlueJ* han d'aparèixer les icones dels dos paquets del projecte: `pract1` i `pract2`.

g) En el projecte `iip`, obrir el paquet `pract2` fent doble clic en la seua icona. Si tot ha anat bé, en la finestra del paquet han d'aparèixer les icones de cadascuna de les classes que conté i una icona d'una carpeta amb el text `<go up>`, que permet tornar a la finestra del projecte `iip`.

h) En el paquet `pract2`, organitzar/ordenar les icones de les classes tal i com apareixen en la Figura 1; per a això només cal **situar el cursor sobre cada icona i "arrastrar-la" amb la ma que apareix en eixe moment sobre ella fins la posició que ocupa en la figura**.

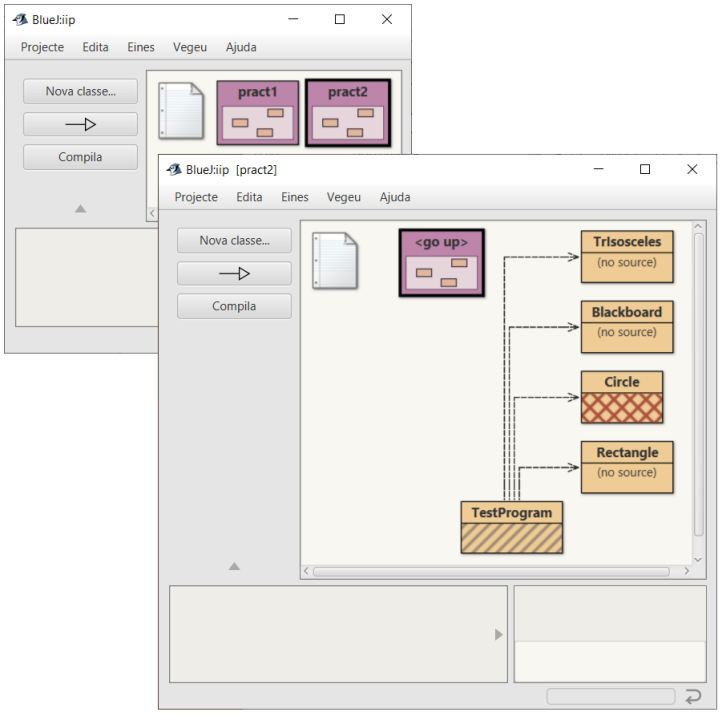


Figura 1: Projecte `iip` que inclou els paquets `pract1` i `pract2` i classes del paquet `pract2`.

i) Guardar el paquet `pract2` **seleccionant l'opció** `Projecte - Desa` **del menú de la finestra del paquet**, per tal que la distribució de les classes en el paquet es mantinga sempre tal i com està.

És important que observes ara les diferents icones de les classes del paquet `pract2` que apareixen a la Figura 1: els associats als fitxers `.java` descarregats apareixen ratllats perquè encara no han estat compilats; els associats als fitxers `.class` descarregats apareixen marcats amb el text `(no source)` per indicar que en el directori corresponent al paquet `pract2` no existeix el codi font (*source*) de la classe i, per tant, aquesta classe sí que es pot fer servir **però no es pot editar o compilar**. D'altra banda, s'observa també com les fletxes entre les classes del paquet indiquen les relacions d'ús que hi ha entre elles.

Operacions d'edició, compilació, execució i comprovació d'estil

Les diferents activitats que es proposen en aquesta secció repassen i amplien l'elenc d'eines *BlueJ* que vas usar en la pràctica 1 per a desenvolupar un projecte i, més concretament, les que concerneixen a les classes que s'ubiquen en un dels seus paquets: opcions d'edició, compilació, execució, i correcció de diferents tipus d'errors.

Activitat #2

a) Editar les classes `Circle` i `TestProgram`, fent doble clic sobre cadascuna de les seues icones o usant l'opció `Obre Editor` del menú de les classes.

b) Comprovar que la primera línia de les classes inclou la directiva del compilador per tal d'indicar que són classes pertanyents a un paquet: `package pract2;`

c) Compilar la classe `Circle`, per exemple des del propi editor, i corregir els possibles errors de compilació que presenta, per exemple, el que es mostra a la Figura 2.

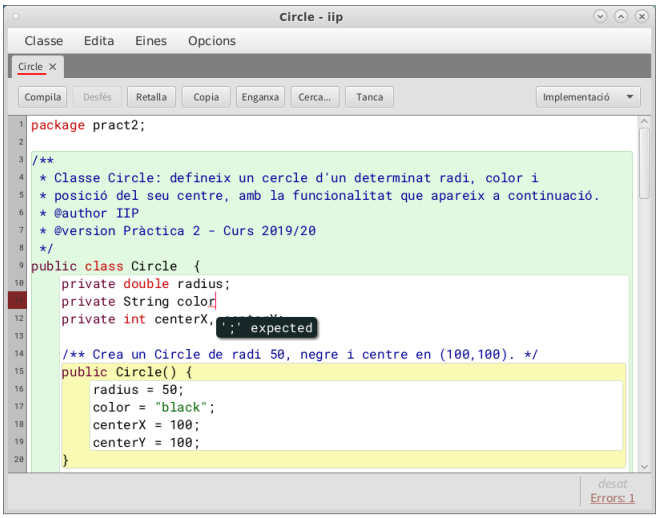


Figura 2: Compilació de la classe `Circle`.

d) Compilar la classe `TestProgram`, por exemple usant l'opció `Compilar` del menú de la classe, i comprovar que no presenta errors de compilació.

e) Comprovar visualment si el codi d'alguna classe presenta errors d'estil i, en eixe cas, corregir-los.

Activitat #3

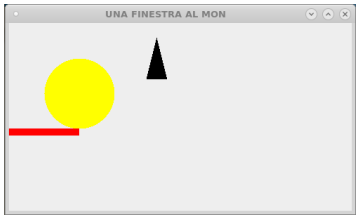
a) Executar el mètode `main` de la classe `TestProgram`. El resultat ha de ser com el que apareix a la Figura 3(c) i en el terminal s'ha de mostrar el missatge de la Figura 4.

b) Comprovar que el missatge que apareix en la finestra de terminal de *BlueJ* NO és igual al que es mostra en la Figura 4. *Si no apareix la finestra de terminal*, s'ha de seleccionar l'opció `Mostrar Terminal` del menú `Vegeu`.

La diferència és deguda al fet que la classe `TestProgram` **té un error lògic**: a l'executar el codi NO es mostra a la finestra de terminal el valor del perímetre del cercle `c` (Figura 4), que era el que pretenia el programador; en el seu lloc, apareix la descripció del cercle `c` (és a dir, `c.toString()`).

(a) Menú de la classe.

(b) Crida al mètode `main`.



(c) Resultat de l'execució.

Figura 3: Execució de la classe `TestProgram`.

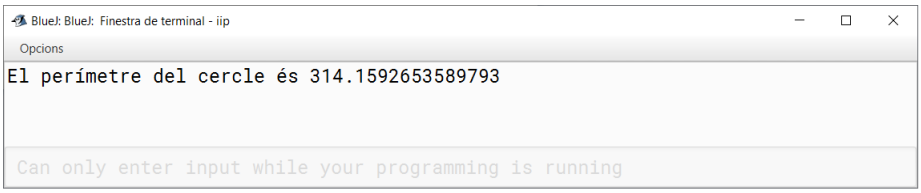


Figura 4: Finestra de terminal de *BlueJ*.

c) Després d'editar la classe, es pot corregir l'error lògic de `TestProgram` completant la instrucció que mostra per pantalla el perímetre de l'objecte `Circle c` *amb ajuda de la funció d'Autocompletat de l'editor de BlueJ*, tal com es mostra a la Figura 5: s'escriu un punt darrere de `c` i es prem `Ctrl-Space`, seleccionant a continuació el mètode correcte dins del llistat que apareix en pantalla (`perimeter()` en aquest cas).

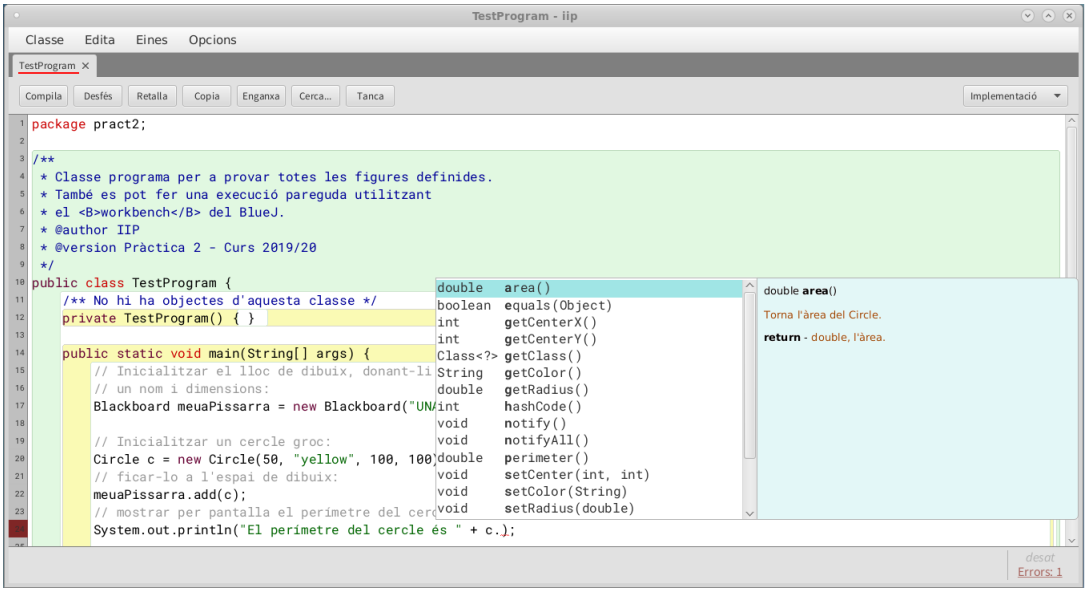


Figura 5: Autocompletat de codi en l'editor de *BlueJ*.

d) Compilar i executar `TestProgram` per tal de comprovar que, una vegada corregit l'error lògic que tenia, en la finestra de terminal ja apareix el perímetre del cercle `c`.

Observa també que a la finestra de terminal es mostren tots els missatges associats a les dues execucions de `TestProgram`, primer l'incorrecte i després el correcte. *Per aconseguir que en cada nova execució s'esborre del terminal el resultat de l'anterior*, s'ha de seleccionar des del menú `Opcions` de la finestra de terminal, l'opció `Neteja la pantalla en cridar el mètode`, com a la Figura 6. Per a futures pràctiques, s'haurà de seleccionar també l'opció `Buffering il·limitat`.

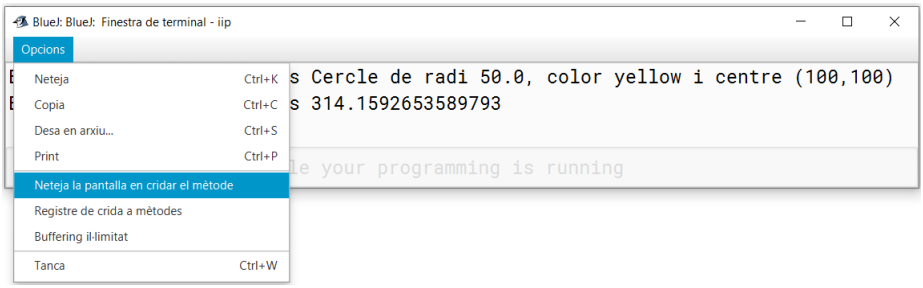


Figura 6: Opcions de la finestra de terminal de *BlueJ*.

És important assenyalar que 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, apareix atenuada, com es pot observar a les Figures 4 i 6.

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 #4

Generar la documentació del projecte i consultar-la. El resultat serà com el que apareix a la Figura 7. Fent "click", per exemple, en l'enllaç de la classe `Circle` es pot consultar la seua documentació.

L'opció *Ajuda*

La utilitat `Llibreries de classes Java` permet accedir a la documentació de l'estàndard de Java. En la instal·lació per defecte aquesta opció és un enllaç a `https://docs.oracle.com/en/java/javase/11/docs/api/` però, 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 `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.

Figura 7: Documentació del projecte `pract2`.

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 8).

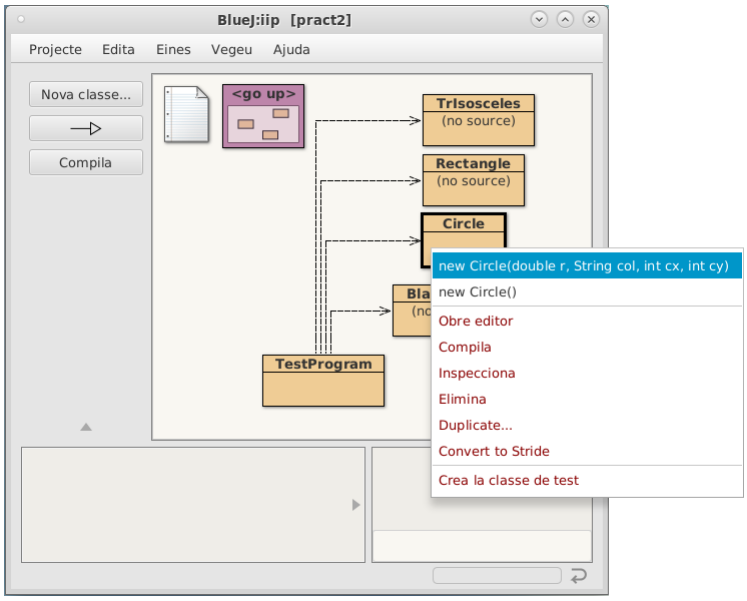
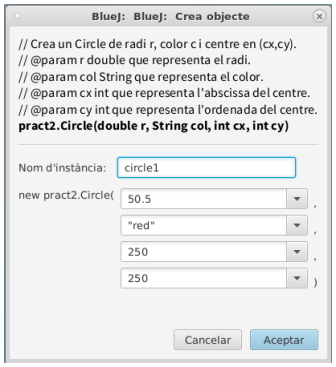


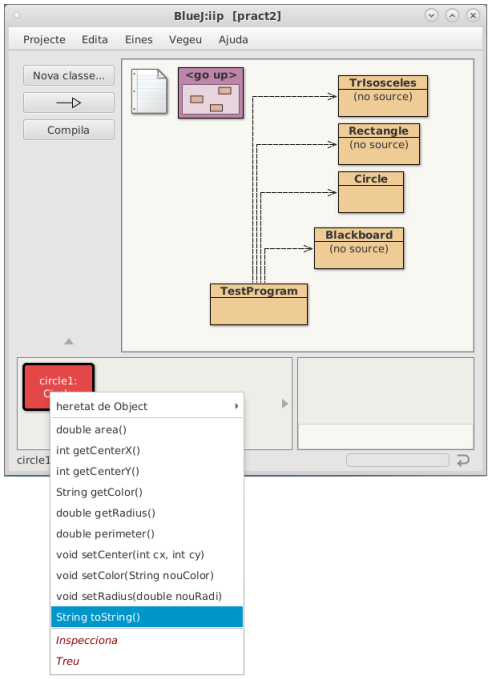
Figura 8: Menú de la classe `Circle`.

Creació d'un objecte

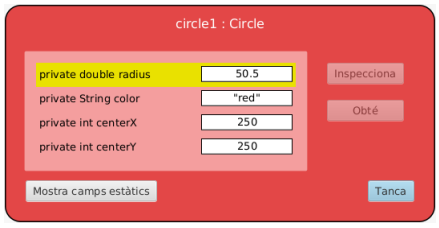
Si es vol crear un objecte s'ha de seleccionar d'aquest menú una de les operacions constructores i seguir el quadre de diàleg que s'obre (Figura 9(a)). En concret, es demana un nom per a l'objecte. Quan es crea aquest objecte, apareix a la part inferior esquerra de la pantalla principal de *BlueJ*, a la zona coneguda com *banc d'objectes* (*Object Bench*) (Figura 9(b)).



(a) Creació d'un objecte `Circle`.



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



(c) Inspecció de l'objecte `Circle`.

Figura 9: Creació d'un objecte en el banc d'objectes de *BlueJ*, menú i observació de l'estat d'aquest objecte.

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 9(b)). Per executar un d'ells només cal seleccionar-lo. Si l'objecte hereta mètodes d'altres classes també apareixen a través de submenús.

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 9(c)).

Activitat #5

a) Crear un objecte de la classe `Circle` de radi 50.5, color "red" i amb centre en $(250, 250)$.

b) Consultar els valors dels atributs de l'objecte de tipus `Circle` creat.

c) Executar el mètode `toString()` definit a la classe `Circle` sobre l'objecte creat.

d) Modificar el radi del `Circle` perquè valga 30.0.

e) Executar novament el mètode `toString()`.

f) Crear un objecte de la classe `Blackboard` amb títol "Dibuix" i dimensió 500 x 500. Per tal que puga veure's l'efecte del següent ítem, l'objecte creat no ha de tancar-se.

g) 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 10). 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.

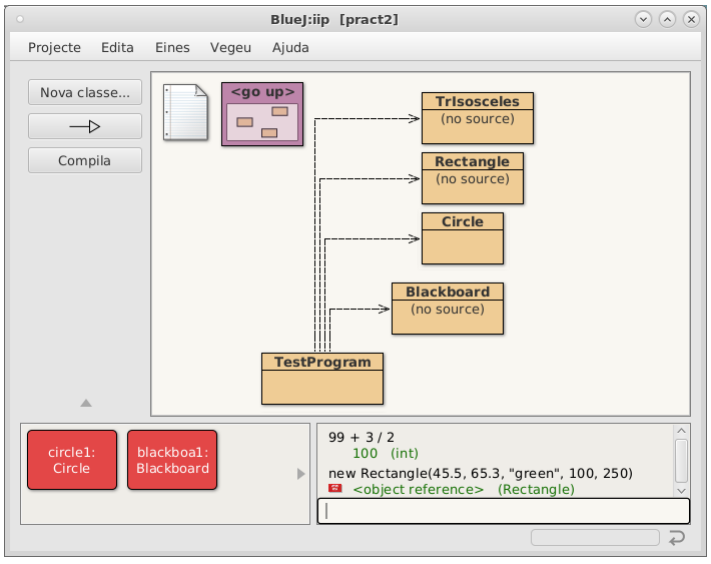


Figura 10: Zona de codi de *BlueJ*.

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 contextual o bé via la zona de codi.

Activitat #6

a) 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`

b) 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:

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

c) 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:

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

d) 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 #5.

e) Executar en la zona de codi el mètode `toString()` sobre `circle1` i arrossegar la icona de l'objecte `String` resultat al banc d'objectes.

f) Crear un nou objecte `Circle` a la zona de codi i arrossegar la seua icona al banc d'objectes. Anomenar-lo `circle2`.

g) Escriure a la zona de codi una expressió que torne el color de `circle2`.

h) Escriure en la zona de codi una instrucció per a afegir `circle2` a la pissarra `blackboa1`, objecte `Blackboard` situat en el banc d'objectes des de l'Actividad #5.

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 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 del punt de ruptura establert, simplement fent clic sobre ella, i després polsar el botó `Continuar` de la finestra del depurador.
3. **Inspecció de variables i comprovar el 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 #7

a) 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`.

b) 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.

c) Per inspeccionar les variables, fer doble clic sobre elles en la finestra del depurador.

A la Figura 11 es mostra el depurador de *BlueJ* una vegada alcançat l'últim punt de ruptura en l'execució de la classe `TestProgram`.

Activitat #8

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

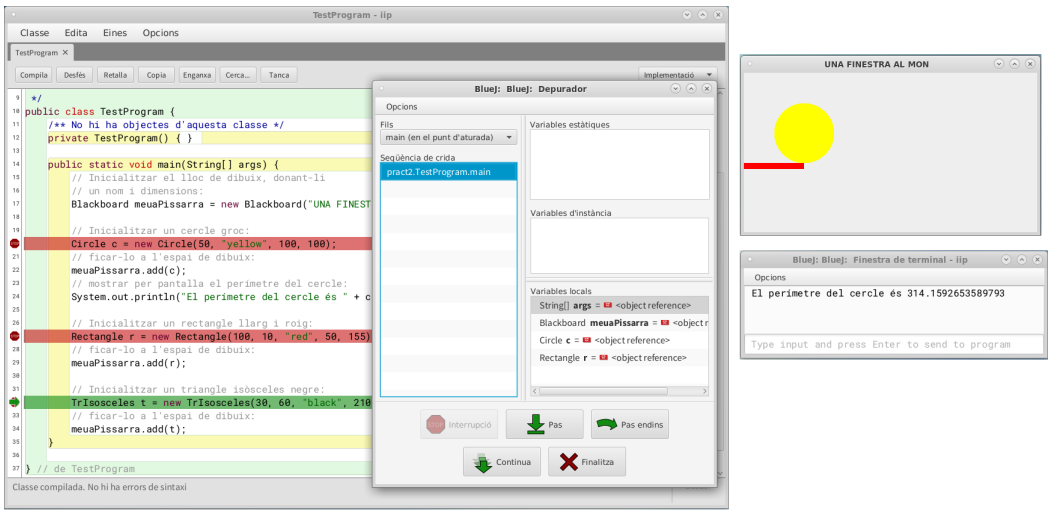


Figura 11: Depurador de *BlueJ*.