Параграф 7

Деякі ключові засоби середовища Pharo

Мета цього розділу полягає в тому, щоб представити основні інструменти середовища програмування Pharo. Ви вже бачили, як визначити класи і методи з використанням браузера; цей розділ покаже вам більше його можливостей, і ознайомить вас з деякими іншими браузерами.

Звичайно, дуже рідко ви можете виявити, що ваша програма працює не так, як ви очікуєте. Pharo має чудовий налагоджувач(дальше «дебагер»), але, як більшість потужних інструментів, він може здатися незрозумілим при першому його використанні. Ми проведемо вас через весь процес налагодження і продемонструємо деякі особливості дебагера.

Однією з унікальних особливостей Pharo (і його предків) є те, що в той час як ви програмуєте, ви живете в світі живих об'єктів, а не в світі статичного тексту програми. Це дозволяє отримати дуже швидку відповідь в той час як ви програмуєте, що робить вас більш продуктивним. Існує інструмент, який дозволить вам подивитися, і дійсно змінити, живі об'єкти: *inspector.*

# **Огляд середовища Pharo**

***System Browser*** є основним інструментом розробки. Ви будете використовувати його щоб створити, визначити і організувати класи і методи. З його допомогою ви можете також перейти через всі класи бібліотеки. На відміну від інших середовищ, в яких вихідний код зберігається в окремих файлах, в Pharo всі класи і методи містяться в системі.

Інструмент ***Finder*** дозволить вам знайти методи, класи, прагми і багато іншого. Ви можете шукати ім'я методів, ім'я класів, деякий вихідний код, імена прагм або навіть шукати методи, надаючи приклад.

***Monticello Browse*r** є початковою точкою для завантаження коду, і збереження коду з та в пакети ***Monticello***. Це обговорюється більш детально в розділі: *спільне використання коду і управління версіями.*

***Process Browser*** надає відображення всіх процесів(потоків), які виконуються у *Smaltalk.*

***Test Runner*** дозволяє тобі запускати і налагоджувати SUnit тести. Він є детальніше описаний у розділі *sUnit*.

***Transcript***це вікно для вихідного потоку Transcript, який є корисним для написання логів.

***Playground***це вікно у котре ти можеш щось вводити. Воно може бути використане з будь-якою ціллю, але найчастіше використовується для написання виразів Pharo та виконання їх за допомогою *Do it*. Ми вже коротко ознайомились з Playground(і Transcript) у розділі: Швидкий екскурс Pharo.

***Debugger***має очевидну роль, але з часом ти зрозумієш, що він відіграє ще важливішу, порівняно з дебагерами інших мов програмування, тому що у Pharo ти можеш *програмувати* прямо у дебагері. Дебагер не запускається з меню; зазвичай він відкривається при помилці тестів, при виконанні команди CMD-d. щоб перервати процес виконання, чи вставити *Halt now* вираз у код.

**Групи вікон**

Керування кількома вікнами в Pharo може стати виснажливим процесом. ***Window Groups*** це вікна, які надають можливість вкладок, аналогічну тій, яку ви звикли використовувати на вашому веб-браузері. Щоб створити групу, натисніть на стрілку вниз, яка з'являється у верхньому правому куті кожного вікна в Pharo і виберіть *Create window group*. Це перетворить це вікно в *window group* з панеллю вкладок, що містить в якості своєї першої вкладки вміст цього вікна. Ви можете додати інші вікна до групи (звичайно, тільки в межах Pharo), шляхом перетягування їх заголовків на панель вкладок, поруч з існуючими. Кожне нове вікно, яке ви перетягуєте на панель вкладок буде додане як нова вкладка.

**Теми і набори іконок**

Якщо, як і деякі з нас, ви не любитель яскравої теми за замовчуванням, то є також темна тема, яка може бути використана в Pharo. ЇЇ можна знайти в *Settings browser* в *World Menu* (*World* > *System* > *Settings*), в розділі *Appearance*. Випадайка *User interface* дозволяє перемикатись між темою за замовчуванням *Pharo3* і темою *Pharo3 Dark*. Ви також можете експериментувати з наявними наборами іконок в наступному випадаючому меню в цьому розділі.

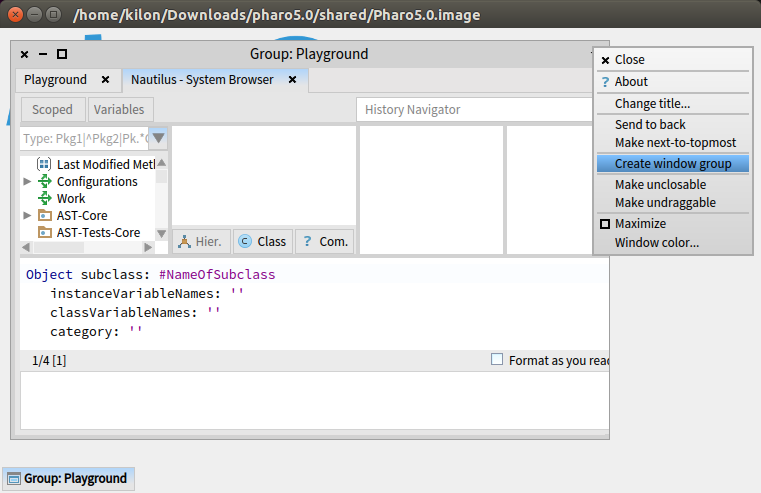
Будь ласка, зверніть увагу, що вам, можливо, доведеться закрити і знову відкрити існуючі вікна, щоб перемальовувати їх правильно з новою темною темою.

Рис 7.1. Групове вікно з двома вкладками

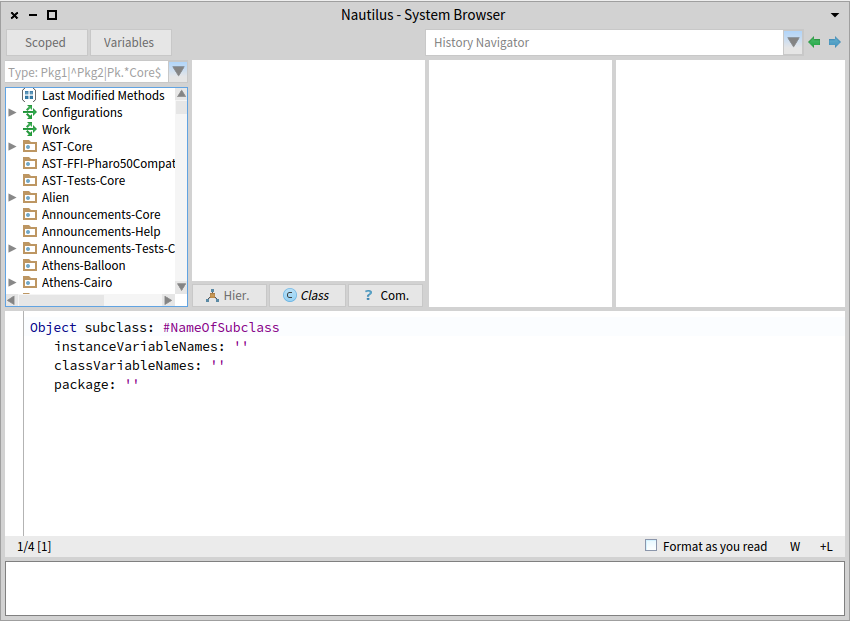
одна з Playground, а інша з System Browser

Ви можете використовувати *Catalog Browser* щоб встановити додаткові теми. Відкрийте *World > Tools > Catalog Browser* і пошукайте теми в полі пошуку. Наприклад, якщо ви хочете щось більш барвисте, ви можете завантажити тему *Nireas*, яка є синьою темою, подібною класичним домашнім комп'ютерам, як *Amiga 500* і *Amstrad CPC 6128*. *Nireas* поставляється з графічним інтерфейсом, який дозволяє вам налаштувати кольори по своєму смаку, якщо синій вам не подобається.

Звичайно, якщо ви дуже хочете пригод, ви можете навіть зробити свою власну тему, використовуючи існуючі, в якості шаблонів.

# **Браузер основного коду**

Багато різних браузерів класів були розроблені протягом багатьох років для Pharo. Pharo спрощує цю історію, пропонуючи один браузер, який об'єднує різні представлення. Малюнок 7.2 демонструє як виглядає браузер, коли ви тільки його відкрили.

 Рис 7.2 Основний браузер коду

Чотири маленькі панелі у верхній частині браузера являють собою ієрархічний перегляд методів в системі, так само як в *Mac OS X Finder* ми можемо переглядати список файлів на диску. Ліва панель перелічує пакети(*packages*) класів; виберіть один (скажімо *Kernel*) і одразу панель справа буде показувати всі класи в цьому пакеті.

Точно так само, якщо ви вибираєте один з класів в другій панелі, скажімо, *Boolean* (Див Малюнок 7.3), то третя панель покаже всі протоколи(*protocols*), визначені для цього класу, а також віртуальний протокол *--all--.* Протоколи є способом категоризації методів; вони допомагають легше шукати методи і уявляти поведінку класу, розбиваючи його на більш дрібні, концептуально когерентні частини. Четверта панель показує імена всіх методів, визначених в обраному протоколі. Якщо потім ви виберете ім'я методу, вихідний код відповідного методу відобразиться в великій панелі знизу, де ви зможете побачити його, редагувати та зберегти відредаговану версію. Якщо ви виберете клас *Boolean*, протокол *controlling*, і метод *or:*, браузер має мати вигляд як на рисунку 7.4.

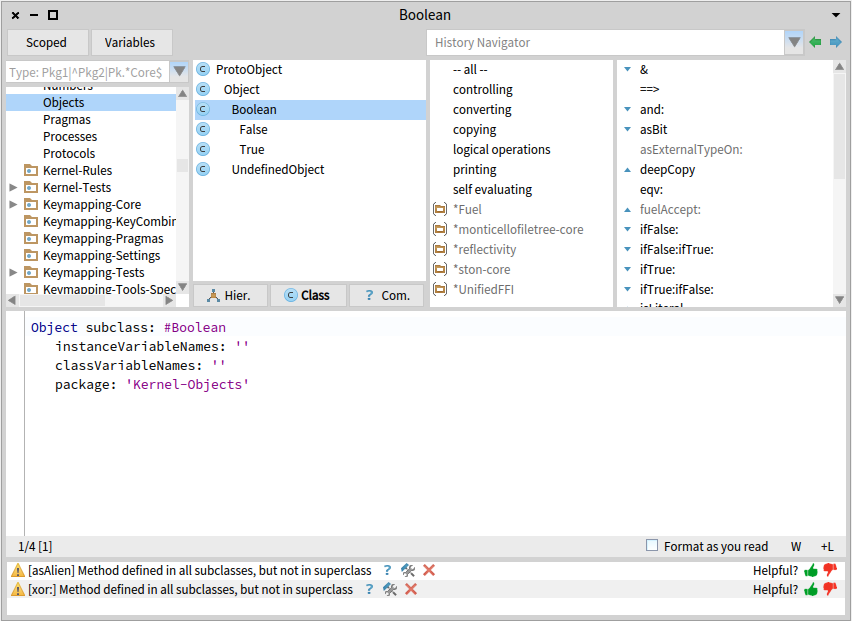
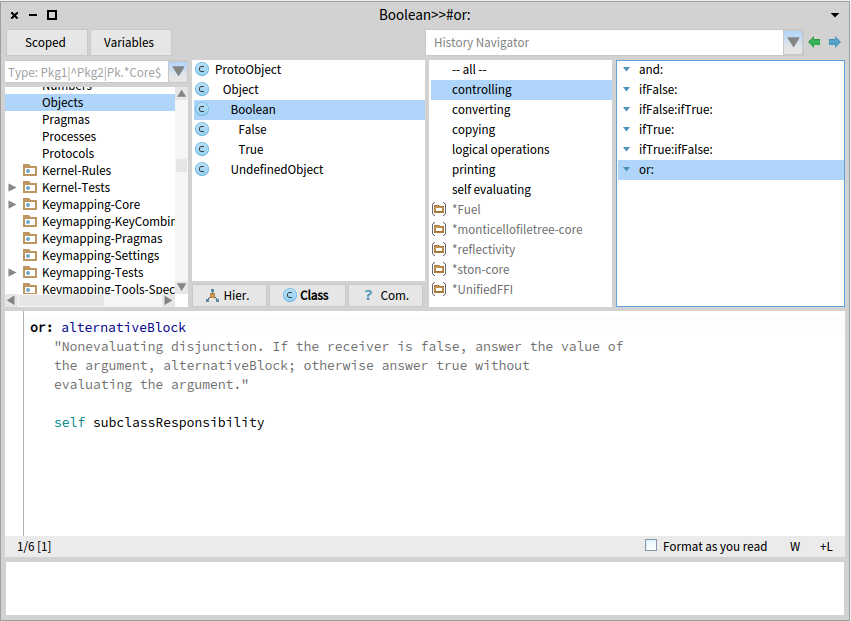
 Рис 7.3 Браузер з вибраним класом Boolean

Рис 7.4 Метод or: класу Boolean

На відміну від каталогів в файловому браузері, чотири верхніх панелі браузера не є абсолютно рівні. У той час як класи і методи є частиною мови *Smalltalk*, пакети і протоколи ні: вони є зручність внесеною в браузер, щоб обмежити кількість інформації, яка повинна бути показана на кожній панелі. Наприклад, якщо б не було ніяких протоколів, браузер мав би показати список усіх методів в обраному класі; для багатьох класів цей списку був би занадто великим, щоб зручно переміщатися по ньому.

Через це, створення нового пакету чи нового протоколу відрізняється від створення нового класу або нового метод. Щоб створити новий пакет, клацніть правою кнопкою миші на панелі пакетів і виберіть *Add package* .... Щоб створити новий протокол, клацніть правою кнопкою миші на панелі протоколів і виберіть *Add protocol*.... Введіть ім'я нового об’єкта в діалозі, і все готово: в пакета чи протокола немає нічого більше, крім його імені та його змісту.

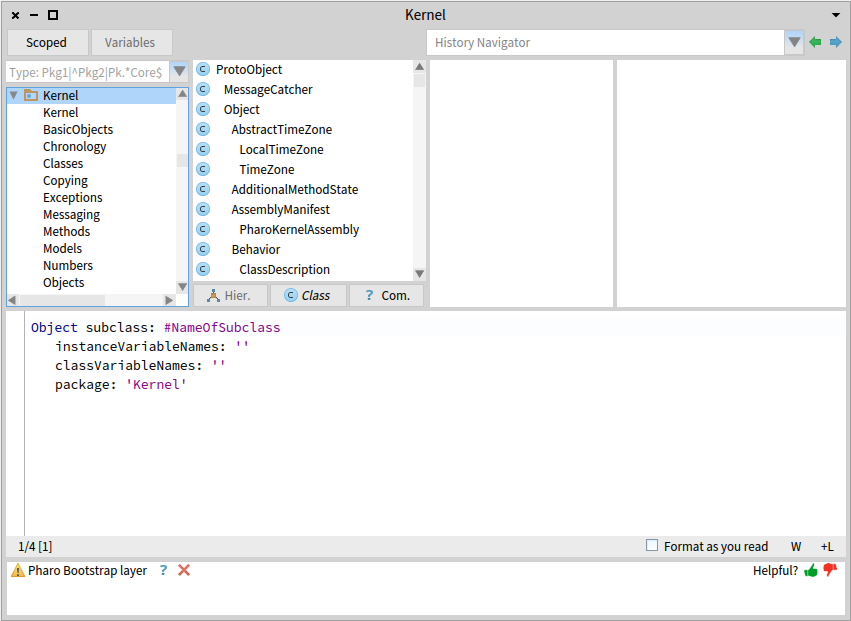
На відміну від цього, щоб створити новий клас або новий метод, ви дійсно повинні написати деякий *Smalltalk* код. При натисканні на обраний пакет (в крайній лівій панелі), в нижній панелі браузера буде відображатися шаблон створення класу(Рисунок 7.5). Ви створюєте новий клас, редагуючи цей шаблон: замінюючи *Object* ім’ям існуючого класу, від якого ви хочете створити свій підклас, замінюючи *NameOfSubclass* ім’ям, яке ви хочете дати своєму новому класу, і заповнюєте назви змінних для екземплярів цього класу.

Рис 7.5 Шаблон створення класу

Пакетом для нового класу за замовчуванням є обраний пакет, але ви можете змінити його теж, якщо хочете. Якщо у вас вже є вибраний певний клас, від якого ви хочете унаслідувати свій, ви можете отримати той же шаблон з трохи іншою ініціалізацію, клацнувши правою кнопкою миші в панелі класів і вибравши пункт *Add Class*. Ви також можете просто змінити визначення існуючого класу, змінити ім'я класу на якесь нове. Завжди, коли ви приймаєте нове визначення, новий клас (той, чиє ім'я слідує за #) створюється (як відповідний метаклас). Створення класу супроводжується створенням глобальної змінної, яка посилається на клас, саме тому ви можете звернутися до всіх існуючих класів використовуючи їх імена.

Чи знаєте ви чому ім'я нового класу має бути як *Symbol* (тобто з префіксом #) в шаблоні створення класу, але після того, як буде створено клас, код може доступатися до класу, використовуючи ім'я в якості ідентифікатора (без #)?

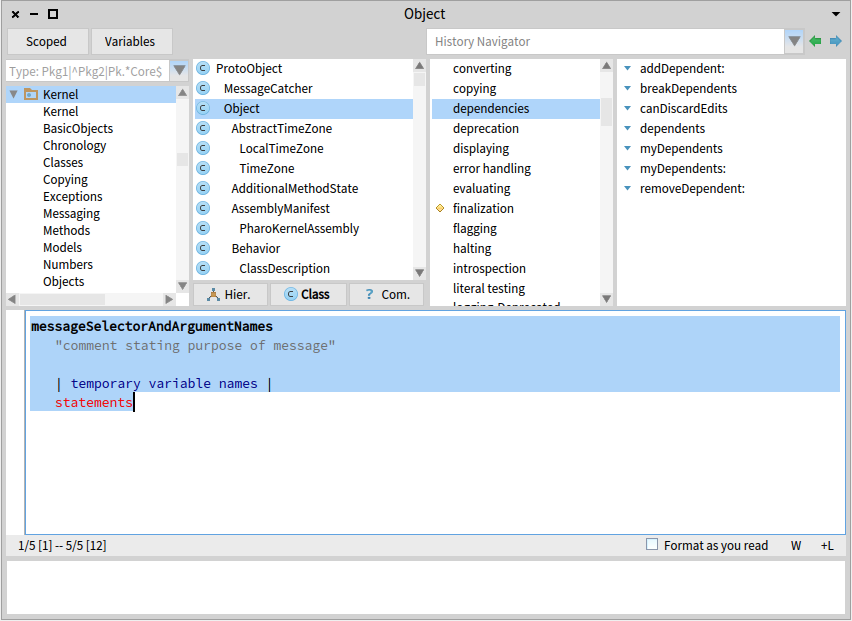
Процес створення нового методу аналогічний. Спочатку виберіть клас, в якому ви хочете додати метод, а потім виберіть протокол. Браузер відобразить шаблон створення методу, як показано на малюнку 7.6, який ви можете заповнити чи змінити.

Рис 7.6 Шаблон створення методу

**Переміщення по просторах коду**

Браузер надає кілька інструментів для вивчення і аналізу коду. Ці інструменти можна отримати, клацнувши правою кнопкою миші в різних контекстних меню, або, у випадку найбільш часто використовуваних інструментів, за допомогою гарячих клавіш.

Відкриття нового вікна браузера

Іноді ви хочете відкрити кілька вікон браузера. Коли ви пишете код, майже завжди вам буде потрібно принаймні два: один для методу, який ви пишете, і інший, щоб переглядати в системі, як все працює. Ви можете відкрити клас з ім'ям будь-якого обраного тексту за допомогою сполучення CMD-b.

**Виконати** Спробуйте це: У вікні *Playground* введіть ім'я класу (для прикладу *Morph*), виберіть його, а потім натисніть *CMD-b*. Цей прийом часто буває корисним; це працює в будь-якому текстовому вікні.

Відправники повідомлень

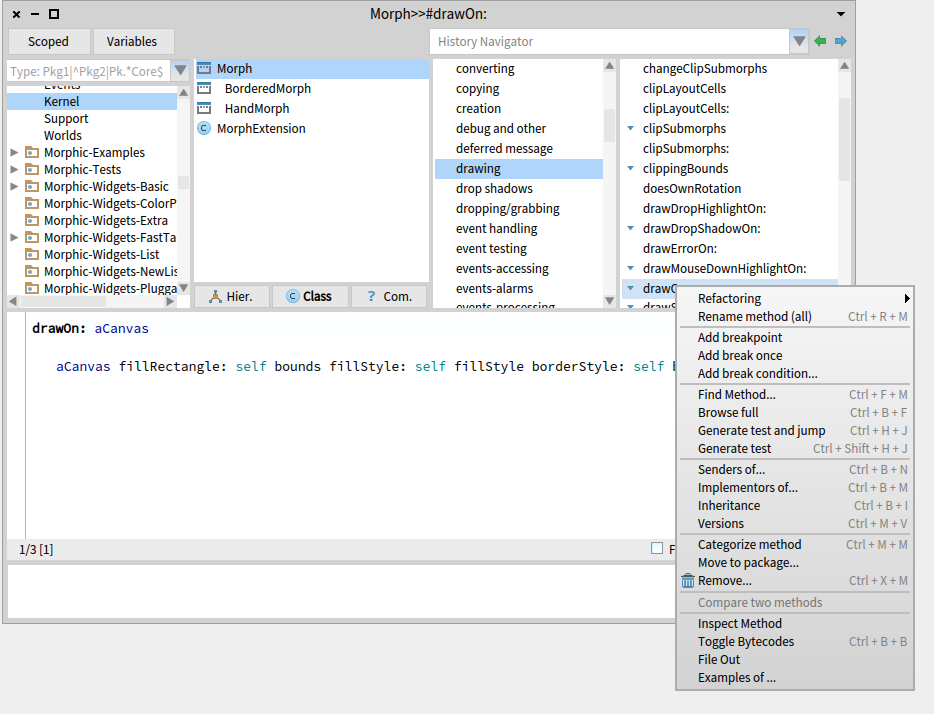
При написанні нового коду, рефакторингу існуючого коду, або при спробі дізнатися як використовувати незнайомі бібліотеки, ви часто будете хотіти знати відправників та імплементаторів різних повідомлень.

Рис 7.7 Опція Senders Of...(b,n)

Є кілька способів виявлення, де в коді використовується повідомлення, шляхом перерахування його відправників:

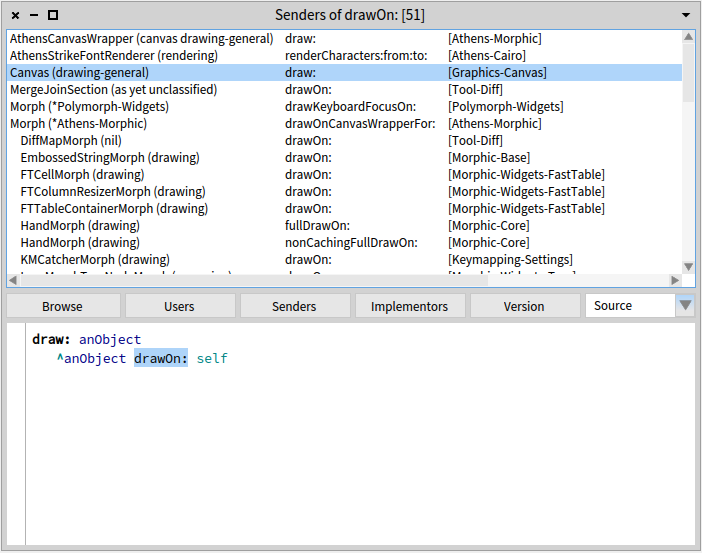
1. **В панелі методів**. Виберіть метод в панелі методів браузера. Ви можете натиснути правою кнопкою миші на ній і вибрати *Senders of...(b,n)* в контекстному меню. Крім того, ви можете також використовувати гарячі клавіші *CMD-b CMD-n* щоб зробити те ж саме (це те, що означає *b,n* в пункті меню). Щоб запам’ятати ці команди пам’ятайте: ***b****rowse se****n****ders.*
2. **На панелі коду**. Виділіть конкретне повідомлення у вихідному коді. Це може бути зроблено в вікні коду, в вікні *Playground*, або в будь-якому текстовому вікно. Якщо ви хочете знайти відправників повідомлення, виділіть всі ключові слова в повідомленні, в тому числі аргументи. Потім, ви можете натиснути правою кнопкою миші на виділеному селекторі і вибрати *Code search...> senders of it (n).* Крім того, ви можете використовувати гарячі клавіші *CMD-n* замість правої кнопки миші.
3. **Використовуючи *Spotter*.** Внесіть конкретний метод в *Spotter* (натисніть SHIFT-Enter, щоб відкрити вікно пошуку *Spotter*, введіть

Рис 7.8 System Browser демонструє що Canvas>>draw метод надсилає drawOn: повідомлення своїм аргументам

селектор повідомлення, стрілку вниз до певного реалізатора цього повідомлення і натисніть СMD-стрілка вправо, щоб сфокусувати пошук на ньому.) Список відправників тепер з'являється в результатах пошуку.

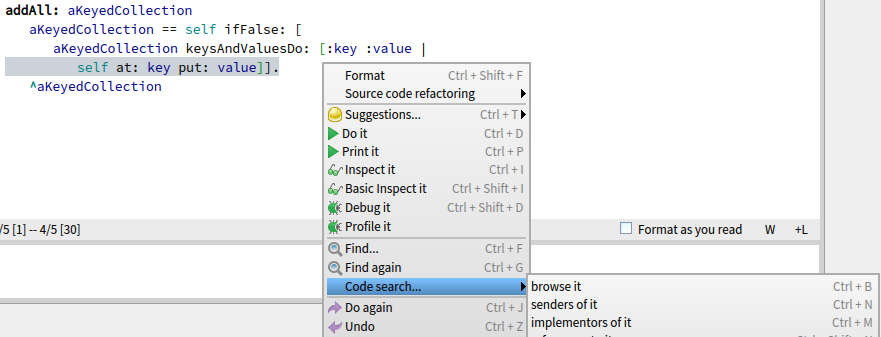


Рис 7.9 Пошук відправників повідомлення у панелі коду

Лише кілька відправників відображаються за замовчуванням, але ви можете переглянути повний список натиснувши на стрілку біля категорії *Senders* (або стрілка вниз до *Senders* і розширити його, натиснувши *CMD-Shift-стрілка вправо*).

Давайте спробуємо деякі з них в дії.

Відкрийте браузер на класі *Morph*, виберіть *Morph* >> *drawOn*: метод з панелі методів. Якщо тепер натиснути *CMD-b CMD-n* (або правою кнопкою миші в панелі методів і вибрати *Senders of...* (Малюнок 7.7)), браузер відкриється зі списком всіх методів, які посилають обране повідомлення (Малюнок 7.8).

Тепер подивимося на третього відправника в списку, *Canvas>>draw:*. Ви можете бачити, що цей метод посилає *drawOn*: до будь-якого об’єкта, що передається як аргумент, який потенційно може бути екземпляром будь-якого класу взагалі. Потоковий аналіз може допомогти з'ясувати клас приймача деяких повідомлень, але в цілому, немає простого способу для браузера, щоб дізнатися, яке повідомлення, могло спричинити виклик певного методу. З цієї причини, браузер *Senders* браузер точно показує, що передбачає його назва: всі відправники повідомлення з обраним селектором. Браузер відправників є дуже корисним коли вам потрібно зрозуміти, як ви можете використовувати метод: він дозволяє швидко переміщатися по прикладах використання. Так як всі методи, з тим же самим селектором слід використовувати таким же чином, всі використання даного повідомлення повинні бути подібними.

**Виконати** Перейдіть до класу *Dictionary* в браузері (пам'ятаєте, що ви можете натиснути правою кнопкою миші в пакеті або панелі класів і вибрати *Find class...*, або просто використати команду CMD-f CMD-c) і виберіть *addAll:* метод в панелі методів.

Реалізатори повідомлень

Крім того, ви можете зіткнутися з повідомленням, і захотіти побачити, як воно реалізовано. Це те, що для чого призначений браузер *Implementors*. Він працює так само як браузер *Senders*, але перераховує всі класи, які реалізують метод з тим же самим повідомленням.

1. **В панелі методів**. Виберіть метод з панелі методів. Ви можете викликати браузер *Implementors*, клацнувши правою кнопкою миші на методі і вибір *Implementors of...(b,m)* в контекстному меню (або використати поєднання клавіш *CMD-b CMD-m*). Щоб запам'ятати сполучення: ***b****rowse i****m****plementors.*
2. **На панелі коду**. Виділіть конкретне повідомлення у вихідному коді (або будь-якому текстовому вікні). Якщо ви хочете знайти реалізаторів повідомлення, ви виділяєте всі ключові слова в повідомленні. Потім, ви можете натиснути правою кнопкою миші на виділеному і вибрати *Code search... > implementors of it (m)* з меню (або просто використовувати команду *CMD-n*).
3. Використовуючи *Spotter*. Внесіть метод в *Spotter* (натисніть *SHIFT-Enter*, щоб викликати вікно пошуку *Spotter*, і почніть вводити селектор повідомлень). Категорія *Implementators* відобразиться в результатах пошуку, показуючи першу найбільш підходящу реалізацію. Щоб побачити повний список, клацніть на стрілку праворуч від категорії *Implementator* (або стрілка вниз і натисніть *SHIFT-CMD- стрілка вправо*).

Спробуйте це: Натисніть *SHIFT-Enter* і введіть *drawOn:* в поле пошуку *Spotter*. Ви повинні побачити список, який показує 5 з 100 реалізаторів цього методу. Це не повинно бути дивним, що так багато класів мають реалізацію цього методу: *drawOn:* це повідомлення, яке розуміється кожним об'єктом, який здатний малювати себе на екрані.

Зверніть увагу, що якщо ви набрали тільки *drawOn* і без двокрапка (:), кількість реалізаторів в результаті пошуку більша. Це відбувається тому, що *Spotter* робить частковий пошук, в тому числі будь-яких методів, які мають «drawOn» в назві, такі як *drawOn: offset:*, *drawOnAthensCanvas*:, і так далі. Це корисно, якщо ви хочете знайти метод, але можеnt згадати тільки частинe його імені.

Наслідування та перевантаження методів

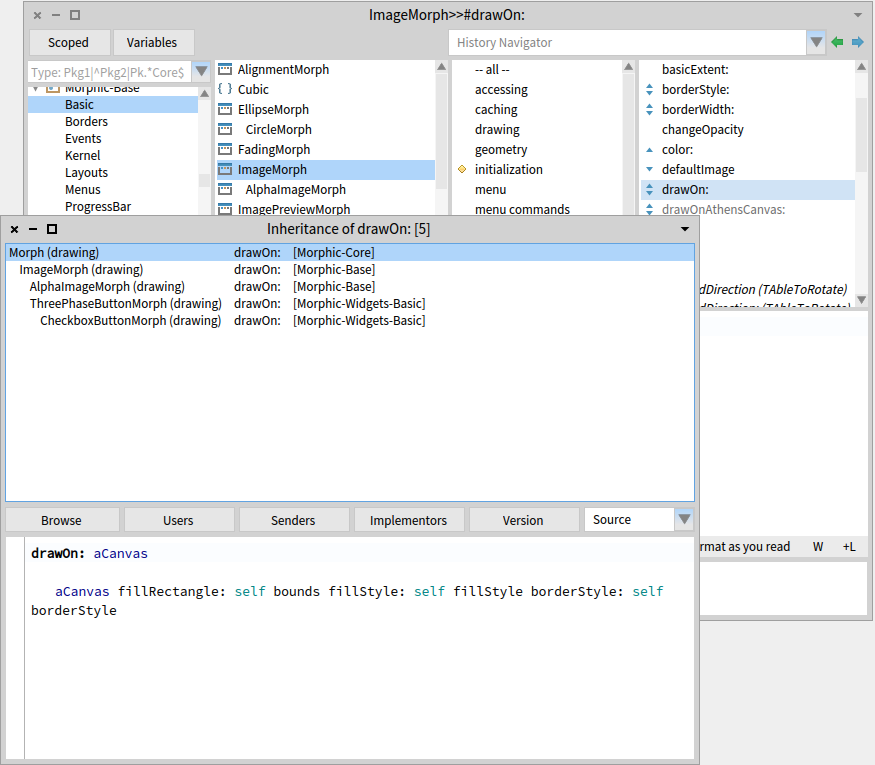
Браузер наслідування відображає всі методи які перевизначають вибраний метод. Щоб побачити, як він працює, виберіть *ImageMorph >> drawOn:* метод в браузері. Зауважте стрілку поруч з ім'ям методу (рис 7.10). Стрілка вгору говорить вам, що *ImageMorph >> drawOn :* перевантажує успадкований метод *(Morph >> drawOn :),* а стрілка вниз вказує вам, що він буде перевизначений підкласами. (Ви також можете натиснути на іконку для переходу до цих методів.) Тепер клацніть правою кнопкою миші на ньому в панелі методів і виберіть *Inheritance*.. Браузер успадкування показує ієрархію методів (рис 7.10).

Рис 7.10ImageMorph>>drawOn: та ієрархія класів, які його перевизначають

Відображення ієрархій

За замовчуванням браузер відображає список пакетів в крайній лівій панелі. Проте можна перейти до режиму відображення ієрархії класів. Просто виберіть конкретний клас, для прикладу *ImageMorph*, а потім натисніть на кнопку *Hierarchy* (*Hier*). Після цього ви побачите в другій панелі ієрархі. класів, що відображає всі батьківські класи і підкласи обраного класу.

Зверніть увагу на те, що панель пакетів заблокована, пакети неактивні. Коли ви в режимі перегляду ієрархій, ви не можете змінювати пакети. Щоб могти змінювати їх знову, закрийте перегляд ієрархій, клацнувши на *Hierarchy* кнопку ще раз.

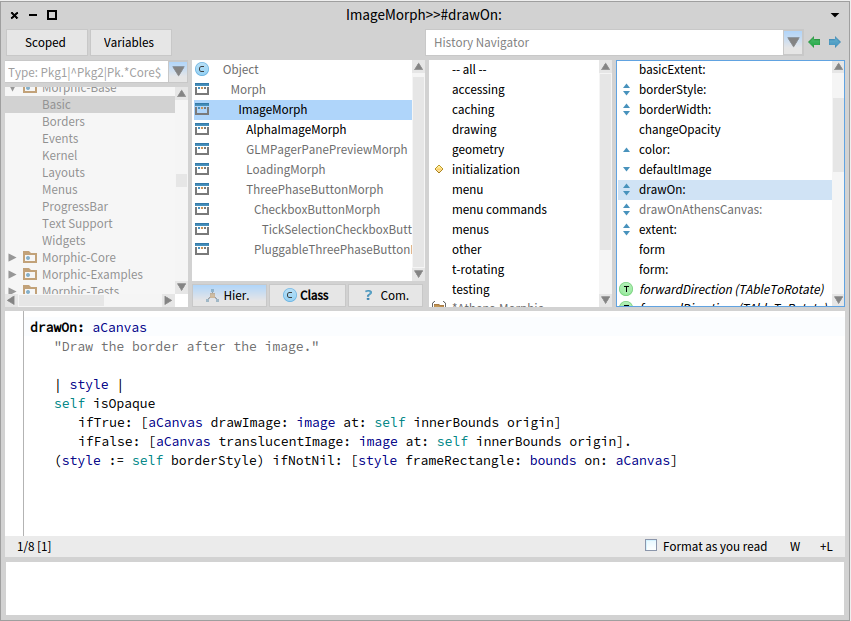
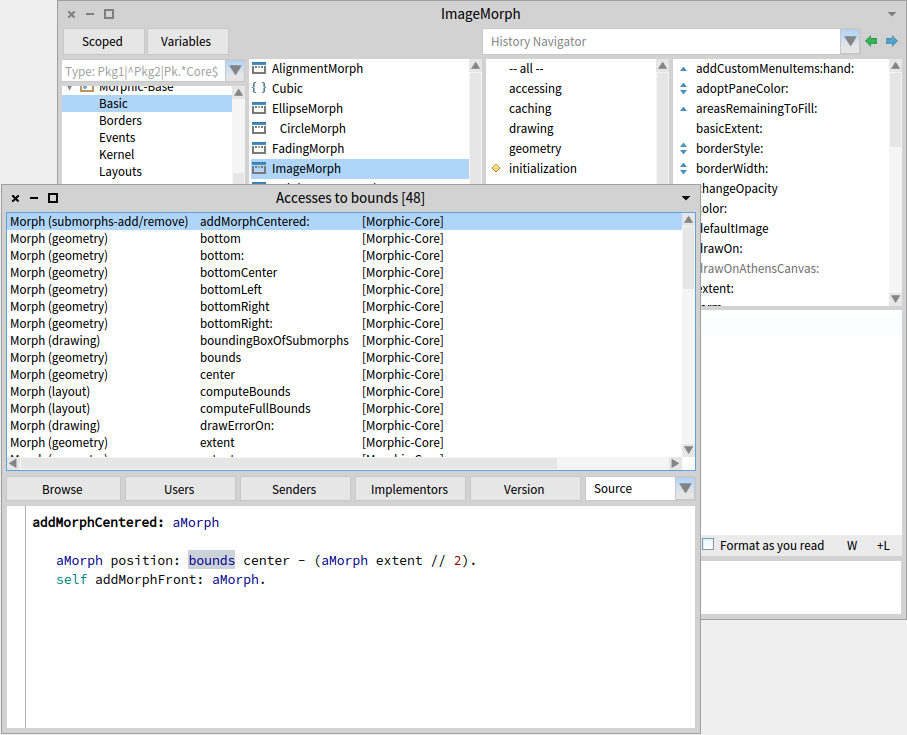
На малюнку 7.11, ієрархія показує, що прямим батьком *ImageMorph* є *Morph*.

Рис 7.11 Перегляд ієрархії ImageMorph

Пошук посилань на змінні

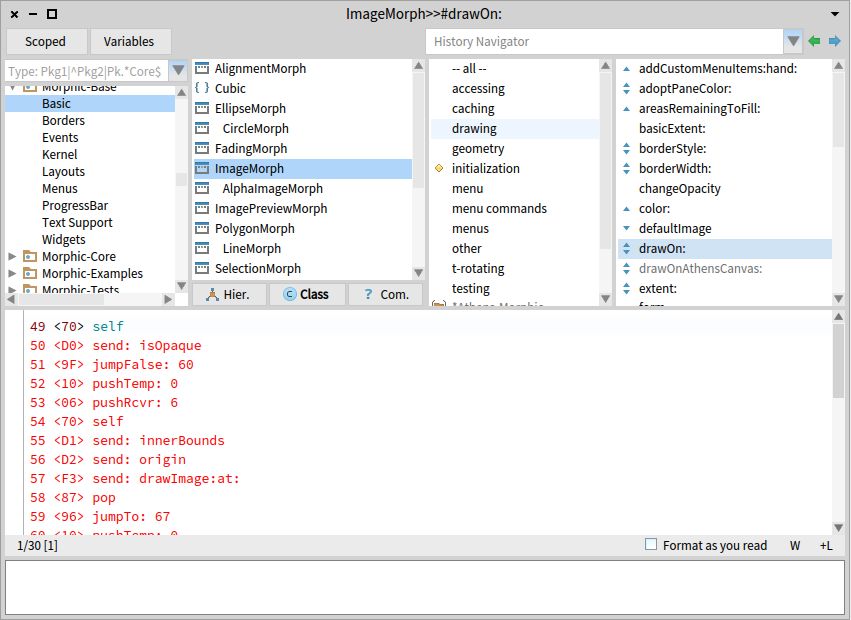
Клацнувши правою кнопкою миші на класі в панелі класів, і вибравши *Analyze > Inst var references...* чи *Analyze > Class var refs...*, ви можете дізнатися де використовуються змінна екземпляру або змінна класу. Ви також можете попасти туди, натиснувши на кнопку *Variables* над списком пакетів. Після того як ви натиснете на кнопку або виберете пункт меню, вам буде представлене діалогове вікно, яке пропонує вам вибрати змінну з усіх змінних визначених в поточному класі, і всіх змінних, які він успадковує. Список є в порядку успадкування; він може бути корисним, просто, щоб нагадати собі назви змінних екземпляра. Якщо клацнути поза списком, то він закриється і ніякого браузера змінних не створиться. Якщо клацнути по змінній, наприклад *bounds*, то створиться *Message Browser*(рис 7,12).

 Рис 7.12 Message Browser для доступу до змінної bounds об’єкта Morph

Ви можете використовувати подібний метод, щоб дивитися на прямі присвоєння змінних (тобто, місця, які модифікують змінні без використання методів доступу). Клацніть правою кнопкою миші на класі і виберіть *Analyze > Inst var assignments.*

Джерело байт-коду

У вас є можливість перегляду байт-коду методу. Щоб зробити це, клацніть правою кнопкою миші на ваш метод, і виберіть *Toggle Bytecodes*, або використовуйте команду CMD-b CMD-b (див малюнок 7.13). Повторно виберіть метод, щоб повернутися до нормального перегляду.

 Рис 7.13 Байт-код метод ImageMorph>>#DrawOn:

Рефакторинг

Контекстне меню пропонує велику кількість стандартних операцій рафакторингу. Просто клацніть правою кнопкою миші на будь-якій з чотирьох панелей, щоб побачити доступні в даний час операції. Дивіться Малюнок 7.14. Рефакторинг раніше був доступний тільки в спеціальному браузері, який називався *refactoring browser*, але він тепер може бути доступний з будь-якого браузера.

**Меню браузера**

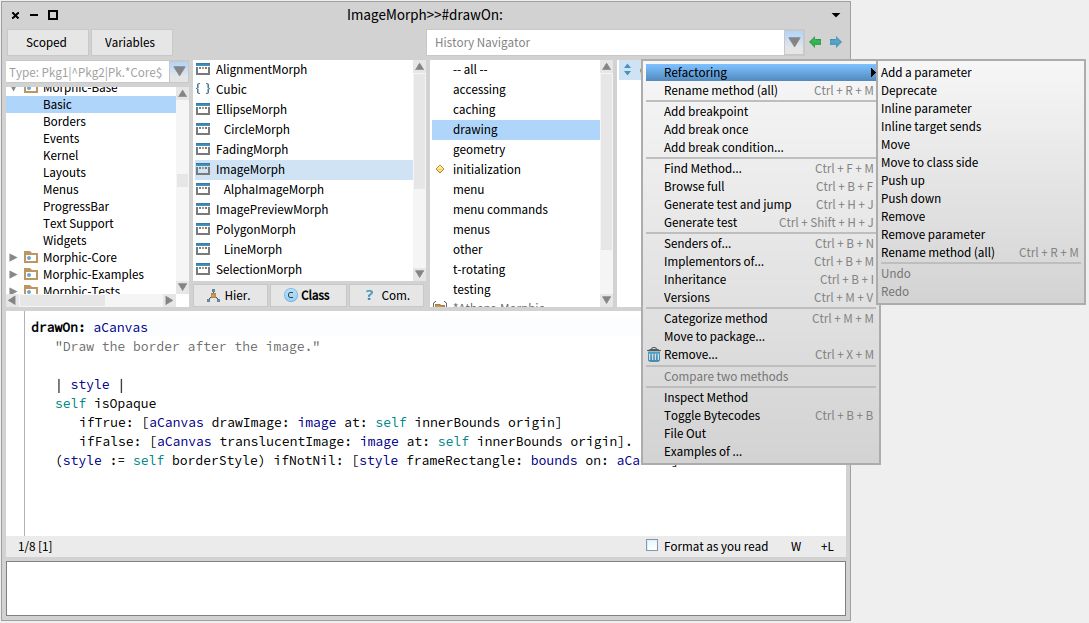
Багато додаткових функції є доступними, клацнувши правою кнопкою миші на панелях браузера. Навіть якщо мітки на пункти меню однакові, їх значення може залежати від конкретних умов. Наприклад, панель пакетів, панель класів, панель протоколів і панель методі, всі мають пункт *File out*, проте вони роблять різні речі: *File out* з пункту меню в панелі пакетів виводить в файл весь пакет, пункт панелі класів виводить в файл методи з усього класу, протокол – все з протоколу, а елемент методу панелі виводить тільки метод який відображається.

Рис 7.14 Операції рефакторингу

Хоча це може здатися очевидним, це може бути джерелом плутанини для початківців.

Мабуть найбільш корисним пунктом є *Find class... (f,c)* в панелях пакетів або класів. Більшість з нас не знає вміст пакетів цілої система, і це набагато швидше набрати CMD-F CMD-С і декілька перших літер назви класу, ніж вгадувати в якому пакеті він може бути.

*Випадайка History Navigator* , яка знаходиться над панеллю протоколів і методів, також може допомогти вам швидко повернутися до класу або методу, який ви недавно переглядали, навіть якщо ви забули його ім'я.

Іншим корисним методом в панелі класів є *Find method (CMD-f CMD-m),* який викликає меню всіх методів в класі і дає вам вікно пошуку.

З іншого боку, якщо ви шукаєте певний метод обраного класу, часто швидше переглядати протокол *--all--*, помістіть курсор в панелі методів, введіть першу літеру імені методу, який ви шукаєте. Це, як правило, буде прокручувати панель так, що шуканий метод буде видимий.

**Виконайте** Спробуйте обидва способи навігації по *OrderedCollection>>removeAt:*

Є багато інших опцій, доступних в меню. Треба потратити трохи часу на роботу з браузером досліджуючи що в ньому є.

**Програмний огляд**

Клас *SystemNavigation* надає ряд службових методів, які є корисними для навігації по системі. Багато з функцій, пропонованих системним браузером реалізуються в *SystemNavigation*.

Відкрийте *Playground* і визначте наступний код для перегляду відправників *drawOn::*

*SystemNavigation default browseAllSendersOf: #drawOn:*

Щоб обмежити пошук відправників серед методів певного класу:

*SystemNavigation default browseAllSendersOf: #drawOn: localTo: ImageMorph*

Оскільки засоби розробки є об'єктами, вони цілком доступні з програм, і ви можете розробити свої власні інструменти або адаптувати існуючі інструменти до ваших потреб.

Програмний еквівалент пункту меню *Implementors of...* є :

*SystemNavigation default browseAllImplementorsOf: #drawOn:*

**Виконати** Дізнатись що є ще доступним, дослідити клас *SystemNavigation* в браузері.