

## **CHAMELEON** Project

# User Guide (Italian version)

# Integrated Development Environment

http://www.di.univaq.it/chameleon/

Version	Changes	Author(s)
V2.0	This user guide describes the main functionalities offered by the CHAMELEON IDE to developers of adaptable Java applications. This is a short version that summarizes the long version, which is available only in Italian. We are working for producing the long version also in English. This version of the Integrated Development Environment supports the new version of the Hybrid (static and dynamic) Analysis approach. The preprocessor derives the Application Descriptor that, among other information, also lists the calls to external library methods for each alternative. Specifically, for each external method call in the list, the programmer is now supported by the CHAMELEON IDE to specify the worst- case call-context on the base of the application	- Marco Autili, - Paolo Di Benedetto - Paola Inverardi
	Iogic that has been coded. When a device requests an application, during the dynamic phase of the analysis, each library method is then executed by the CHAMELEON Client in the target execution environment, and its actual resource consumption is computed. This consumption is used to specify into the profile the resources consumed by each library method. The advantage of combining the static and the dynamic phase is to avoid to statically perform the expensive inter-procedural call-closure for library methods. In fact, the static inter-procedural call- closure in Java is extremely expensive for even a simple "Hello World" program due to the use of standard libraries (see also the developer guide of the CHAMELEON Mobile Client, Server, and UDDIe Registry, and the guide of the Analyzer).	

#### **Table of Contents**

1.	Installazione	1
	1.1. Installazione del Plug-In ChameleonIDE 1.2. Installazione del Chameleon Programming Model	1 1
2.	Configurazione del Plug-In ChameleonIDE	8
3.	Adaptable Java Project	10
3	3.1. Adaptable Java Model decorators	13
4.	L'Adaptable Java Editor del ChameleonIDE	15
4	4.1. Annotation e Resources Annotation wizard	17
5.	L'Adaptable Java Build	21
6.	Adaptable Java Preferences Pages	24
7.	Appendice A - Adaptable Java Icons	1
8.	Appendice B - Le classi Launch.java e Annotation.java	1

## 1. Installazione

I seguenti paragrafi descrivono i passi necessari alla predisposizione di un ambiente di sviluppo per progetti "Adaptable Java".

#### 1.1. Installazione del Plug-In ChameleonIDE

L'installazione del plugin ChameleonIDE si rivela piuttosto semplice dal momento che si limita alla copia del file it.univaq.di.ChameleonIDE.jar direttamente nella directory /eclipse/plugins dell'applicazione Eclipse utilizzata.

Ad ogni ripartenza Eclipse si accorge automaticamente della presenza dei nuovi plugins rendendoli così immediatamente disponibili.

Qualora invece non si disponesse direttamente del Plug-In in formato .jar ma comunque del source Project (it.univaq.di.ChameleonIDE), la generazione di un Plug-In distribuibile in formato .jar potrà avvenire in qualunque momento con i passi operativi descritti in Fig.1.

🖨 Plug-in Development - it	t. univaq. di. Chamele	onIDE/plugin.xml - Eclipse SDK	JAR Export		X
File Edit Source Refactor N	lavigate Search Projec	t Run Window Help	1AR Elle Specification		
i 📑 • 🔛 🖻 i 🎄 • 🔘	• 💁 • 🗄 😫 🖶	🖨 Export	Define which resources should be exported into the JA	R.	»
Click con il tasto destro d	lel mouse in un punto	Select			B
<ol> <li>qualsiasi del Package E</li> </ol>	xplorer View	Export resources into a JAR file on the local file system.	Select the resources to export:		a
😑 😸 it.univaq.di.ChameleonID	DE		🗉 🕑 🔛 it.univaq.di.ChameleonIDE	Classpath	^
🗄 🌮 src	New Co Isto	Select an export destination:		project	
it.univaq.di.Cl	Go Into	type filter text		initial properties	
🗷 🌐 it.univaq.di.Ch	Open in New Window Open Type Hierarchy	🖃 🧀 General		🗹 🎰 plugin.xml	
it.univaq.di.Ch	Show In	* Ant Buildfiles			
🗷 🌐 it.univaq.di.Ch	Copy	Sreakpoints			
it.univaq.di.Cf	Copy Qualified Name	- File System			
🗷 🌐 it.univaq.di.Ch 🖡	Paste	Verferences	Export generated class files and resources		۲
JRE System Library	K Delete	JAR file (3) Selezionare JAR file	Export all output folders for checked projects		
A Referenced Librari	Build Path	Javadoc	Export java source files and resources		
🗉 🗁 icons	Source Refactor	Poployable features	Export refactorings for checked projects. Select re	efactorings	
templates (2) C	lick su Export	>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>	Select the export dectination:		~
build.properties	A Export	Copse product     Ecopse product     Ecopse product	IAD Silve CulDecuments and Settinger/Davide Detrill/D	oditos)it usiusa di Chamoloos II us	
plugin.properties	Defrech	🗷 🧀 Other	JAK ne: C:/bocuments and Sectings(Fande Pednii(D	estoplicantadanchaneleonite	
••••	Close Project		Options:		
	Assign Working Sets		Add directory entries		
	Run As	(4) Click su Next	Overwrite existing files without warning		
	Debug As			(5) Selezionare i files e le onzioni	
1	P Build Fat Jar Team	(2) < Back Next > Einish Ca		desiderate e premere Finish	
	Compare With	•			
	Restore from Local Hist	tory		k Next > Finish Cancel	
-	PUE TOOIS				
L -	Properties	Alt+Enter			
					_
it.univaq.di.Cham	ieleonIDE			i a 🔝 🧭 🛽	

Fig.1: Generazione del plugin it.univaq.di.ChameleonIDE.

La procedura descritta genera il file it.univaq.di.ChameleonIDE.jar nella directory di destinazione prescelta.

#### 1.2. Installazione del Chameleon Programming Model

Allo scopo di rendere più agevole la successiva fase di configurazione ed utilizzo dell'ambiente "Adaptable Java", per quanto riguarda il Chameleon Programming Model si rendono necessari i seguenti passi volti alla generazione di un unico file .jar "integrato" che risulterà così completo sia delle librerie necessarie allo stesso

Programming Model (jdom.jar e antl.jar), sia delle classi java (Launch.java e Annotation.java) indispensabili invece per il suo utilizzo da parte di applicazioni "esterne" quali il ChameleonIDE.

La generazione di un file .jar integrato presuppone l'installazione del plugin "Fat Jar Eclipse Plug-In" facilmente reperabile dal sito <u>http://fjep.sourceforge.net</u>, che anche in questo caso va semplicemente copiato nella libreria /eclipse/plugins dell'applicazione Eclipse SDK utilizzata (in questo caso è indispensabile una versione di Eclipse completa del PDE - Plug-in Development Environment).

Il Programming Model andrà quindi importato rispettando la struttura di progetto rappresentata in Fig.2 in cui risultano evidenzati sia il nuovo package chameleon.programmingmodel.Launch riservato alle classi precedentemente menzionate che consentono l'utilizzo del prodotto

da parte del ChameleonIDE (e di eventuali altre applicazioni "esterne") sia le librerie necessarie alla stessa applicazione (messe a disposizione nel folder jar).

Gli errori rilevati dal Workbench Eclipse in questa fase (le "x" quadrate in rosso di fianco ai packages e al Project Folder) sono temporanei dal momento che non sono ancora stati definiti i path per l'individuazione delle librerie .jar.

Tornando invece brevemente alle classi java, la classe Launch.java si occupa della gestione delle richieste di attivazione del Programming Model da parte di



Fig.2: Struttura di progetto

applicazioni "esterne", mentre la classe Annotation java riveste particolare utilità nell'ambito dello sviluppo dei progetti "Adaptable Java".

Il codice di entrambe le classi è consultabile in appendice.

Si procede quindi con l'indicazione delle Referenced Libraries di progetto ossia delle librerie antl.jar e jdom.jar precedentemente copiate nella cartella jar (con operazione di import oppure con semplice copia/incolla dal file system) seguendo la numerazione dei passi di Fig.3 e Fig.4.

🖶 Plug-in Development - chameleo	on.programming.mode	UMETA-INF/MAN	IIFEST.MF - Eclipse SDK
File Edit Source Refactor Navigate	Search Project Run Wi	ndow Help	
📬 • 📄 📄   🏇 • 🕥 • 💁 •	i 🖄 🕸 🞯 • i 🖄	🔗 i 🖢 - 🖗	- 🎨 🔶 • 🔿 - 🗄 🖉
📕 Package Explorer 🛛 🕸 Plug-ins	- 8	🚯 chameleon.pro	ogramming.model 🛛
<	> -> @   🖻 🕏 ヾ	🚯 Overvie	w
🖃 耐 chameleon.programming.model			
🖻 🥵 src		General Inform	mation
🗈 📅 chameleon.programmingmo	del.AlternativeFactory	This section des	cribes general information about this
the chameleon programming mo     for the chameleon programming mo	del.Conflict	ID:	chameleon.programming.model
🖨 🖶 chameleon.programmingmo	del.Launch	Version:	1.0.0
🗄 🕖 Annotation.java		Name:	Chameleon Programming Model Plu
🗈 册 chameleon.programmingmo	del.Parsing Click con il	ll Iasto destro del n	nouse in un Studi di L'Aquila
🗄 🖶 chameleon.programmingmo	del.Tool (1) punto quals	iasi del Package	Explorer View
IRE System Library [jre1.6.0_0	Ebow In	olt⊥SPittTM ▶	·
jar		AICEDINICEN	L
antır.jar	🗈 Сору	Ctrl+C plug-in when one	plug-in when one of its classes is loa
	Copy Qualified Name		
	💼 Paste	Ctrl+V	ronments
	🗙 Delete	Delete	num execution environments require
Click su Build Path (2)	Build Path	×	隆 Remove from Build Path
	迠 Import		💸 Configure Build Path
	🛃 Export	L	******
			(3) Click su
			Configure Build Path
	Alt+Enter	lencies Runtime Extensions Exter	

Fig.3: Passi di configurazione del Build Path

🔮 Properties fo	chameleon.	programming.model		X 🗆 🔛
type filter text		Java Build Path	(4) Selezionare I	a scheda Libraries 👌 🎫 👄 🗧
<ul> <li>Resource</li> <li>Builders</li> <li>Java Build Path</li> <li>Java Code Style</li> <li>Java Compiler</li> <li>Java Editor</li> <li>Java Editor</li> <li>Java Editor</li> <li>Plug-in Develop</li> <li>Project Referent</li> <li>Refactoring His</li> <li>Run/Debug Set</li> </ul>	Ge JAR Seler Choose jar arc Choose jar arc Choose jar arc Choose jar arc Choose jar arc Choose jar arc	Source Projects JARs and class folders on to ARS and class folders on to The system Libra Plug-in Depender Ction Chives to be added to the build ( releon.programming.model ar antir.jar jdom.jar (6) Selezion onti.jor e	Libraries & c the build path: ry [jre1.6.0_03] ncies path: ; jdom.jor	Order and Export         (5) Click su Add JARs         Add JARs         Add External JARs         Add Variable         Add Library         Add Class Folder         Edit         Remove         Migrate JAR File
0		(7) сн	ck su OK	(8) Click su OK

Fig.4: Selezione delle librerie .jar esterne.

Si noterà a questo punto in Fig.5 che le librerie antlr.jar e jdom.jar appaiono automaticamente "trasferite" dal folder jar al

folder Referenced Libraries.



Fig.5: Il progetto dopo la configurazione del Build Path.

A questo punto il processo si conclude con la generazione del file .jar "integrato" mediante l'utilizzo del Fat Jar come descritto in Fig.6 e Fig.7.

Il file risultante (chameleon.programming.model\_fat.jar) che compare sotto la directory principale del progetto (Fig.8) andrà quindi copiato nella stessa directory /eclipse/plugins del Plug-In it.univaq.di.ChameleonIDE.

🖨 Plug-in Developm	ent - chameleon.prog	gramming.model/ME
File Edit Source Refa	ctor Navigate Search	Project Run Window
Et  Click Deckage Explo	con il tasto destro nouse in punto iasi del	🕸 🞯 • 🤅 🥭 🔗 •
(10) Pack	age Explorer View	
Chameleon.progr     Src     Src     Srstem I	Open in New Window Open Type Hierarchy Show In	F4 Alt+Shift+W ▶
iar iar iar iar iar	Copy	Ctrl+C
build.propert	💼 Paste 💢 Delete	Ctrl+V Delete
	Build Path Source Refactor	► Alt+Shift+S Alt+Shift+T
	≧g Import ≧ Export	
	Refresh Close Project Assign Working Sets	F5
Selezionare (11) [	Run As Debug As	) 
Build Fat Jar	Build Fat Jar     Team     Compare With     Restore from Local His     PDE Tools      Properties	story
	Propercies	AIC+Encer

Fig.6: Generazione del \_fat.jar (avvio del processo).

•					
Configure Fa Config for proj	t Jar Plug-In ect chameleon.programming.model				
Jar-Name:	chameleon.programming.model_fat.jar Bro	owse	Select fil Files for p	les for Fat Jar roject chameleon.programming.model	
Manifest: Main-Class: Class-Path: ( One-Jar-Expan	select Manifest file       Bro <createnew>       Bro         chameleon.programmingmodel.Launch.Launch       Bro         ✓ merge individual-sections of all MANIFEST.MF files       V         ✓ remove signer files (*.SF) in META-INF       One-JAR         d:      </createnew>	owse	File-List:	<ul> <li></li></ul>	output Add Dir
0	< Back Next > Finish Car	incel		Save Settings	Export AN
	(12) Riportare le opzioni indicate e selezionare Next		0	<back next=""></back>	Einish Cance

Fig.7: Generazione del \_fat.jar (selezione delle librerie).



Fig.8: Il file \_fat.jar generato.

## 2. Configurazione del Plug-In ChameleonIDE

Una volta avviato Eclipse la corretta esecuzione dei precedenti passi di installazione ha immediatamente riscontro dal momento che si nota subito la nuova icona nella toolbar, il cui significato, come quello di tutte

le "Adaptable Java Icons", è riportato in appendice al documento. Tuttavia si rende necessaria una semplicissima fase di configurazione iniziale affinchè possa successivamente avvenire senza problemi l'interazione tra il ChameleonIDE ed il Chameleon Programming Model.

Una volta generato il nuovo progetto Java su cui si andrà a lavorare, occorrerà indicare nelle Referenced Libraries il file chameleon.programming.model\_fat.jar contenente tutte le librerie e le classi necessarie alla fase di "Building".

Naturalmente, proprio perché tale operazione investe la sola fase di Building, essa potrà avvenire anche in una fase successiva senza precludere le altre funzionalità e gli strumenti messi a disposizione dal Plug-In ChameleonIDE.

Prima di procedere con il Building sarà pertanto sufficiente seguire i passi di Fig.9 e Fig.10.



Fig.9: Creazione di un nuovo Java Project.

😂 Java - Eclipse Platform						
File Edit Source Refactor Navigate Search Project Run Window Hel	þ					
: 📬 • 🗄 👜 💣 : 🍫 • 🔕 • 🤮 🕸 🚱 • i 🕭 🖨	🛷 i 🖢 - 🖗	🖨 Properties	for AdaptableJa	vaProject	en en sen al <b>familie d</b> ata	
😫 Package Explor 🙁 🏂 Herarchy 🗖 🗆		type filter text		Java Build Path	(4) Selezionare la s	heda Libraries 🗇 🗢 🔿 🕤
AdaptabieJaveProject  AdaptabieJaveProject  Show In Alt+Shift+W  Copy Qualified Name Paste Ctrl+V  Delete (2) Solezionare Build Path  Euld Path  Euld Path  Euld Path  Configure	rom Build Path Build Path (3)	Resource Buiders Java Build B Java Code B Java Code B Java Code B Java Editor JavaGote Project Rel Run/Debug Task Tegs Validation	Path Style cation erences Settings itory <b>figure Build Path</b>	Image: Source     Image: Property of the source       Image: Image: Image: Source     Image: Image: Source       Image:	jects Mitraries s on the build path: Library [re1.6.0_03] ( rules: No rules defined brary location: (None) s.jar - C1/Programmi,Javalyre - C1/Programmi,Javalyre1.6.0 c1/Programmi,Javalyre1.6.0 c1/Programmi,Javalyre1.6.0 c1/Programmi,Javalyre1.6.0 c1/Programmi,Javalyre1.6.0 s.jar - C1/Programmi,Javalyre1.6.0 provider .jar - C1/Programmi,Javalyre1.6.0 provider .jar - C1/Programmi,Javalyre1.6.0 provider .jar - C1/Programmi,Javalyre1.6.0 s.jar -	rder and Export           5) Premere Add External JARs           Add Egternal JARs           Add Egternal JARs           Add Uariable           J3l(b           0.03(b)           Add Class Folder           avalyre1.           e1.6.0_0           Edd. Class Folder           avalyre1.           e1.6.0_0           Edt           e1.6.0_0
Configure	JAR Selection		n me me me me	an a	? 🛛	
Properties Alt+Enter	Cerca in:	plugins		v 0 🕫	😢 🖽 -	Mgrate JAR Hie
(1) Click con il tasto destro del mouse in un punto qualsiasi del Package Explorer View (* Problems 23 @ Ja Derrors, Owarnings, O infos Description *	Documenti recenti Desktop Documenti Ritoroe del computer	crg.junit_3.8.2 crg.mortbay.je com.iorat.jsch com.iorat.jsch javax.servet.j2 javax.servet.j2 grg.apache.com crg.apache.loc grg.apache.loc grg.apache.loc grg.apache.loc grg.apache.loc	v200706111738 ty-source 5.1.11.v2 tramma (model, fait 5.1.v20070906 0.1.31 p.2.0.0.v20070611791 4.0.v20070611793 imons.el_1.0.0.v2007 imons.el_1.0.0.v2007 imons.el_5.1.7.v2007061 core_3.1.200.v2007 is 2.200.v2007082 pare_3.3.1.r33x_200	00706111724 (6) Selezionare il fi chameleon.pr nella directory 003 706111724 v200706111724 11724 v200706111724 11724 v200706111610 11724 v200706181610 11724 v200706181610 11724 v200706181610	ilo ogramming.model. fatja /eclipse/plugins (7) Click sv Apri	OK Cancel (8) Click su OK
JRE System Library [jre1.6.0_03] - AdaptableJavaProject		Nome file:	chameleon.programm	ning.model_fat	Apri Apri	
	Risorse di rete	∐ipo file:	*.jar;*.zip		Annulla	

Fig.10: Configurazione del chameleon.programming.model\_fat.jar.

## 3. Adaptable Java Project

A questo punto siamo in grado di realizzare un nuovo progetto "Adaptable Java". Ricordando che di fatto l'ambiente "Adaptable Java" ChameleonIDE non è altro che una "estensione" del Java Development Environment già fornito dalla piattaforma Eclipse, generalmente si presuppone che il primo passo sia volto alla definizione di una nuova classe Java o, meglio, di una classe Java "Adaptable". La generazione di una nuova classe "Adaptable" può avvenire in diverse modalità:

- Direttamente dalla toolbar con l'azione associata in precedenza;
   all'icona
- Dalla barra di menù selezionando in sequenza File, New, Other..., quindi "Adaptable Class" sotto la nuova categoria "Adaptable Java Category" (vedi Fig.11);
- Eseguendo un click con il tasto destro del mouse in qualunque punto della Package Explorer View in modo da richiamare il pop-up menù di Fig.12 e procedere con gli stessi passi descritti al punto precedente;

🖨 Java - Eclipse Platform				
File (1) Source Refactor Navigate	Search Project	Run Window Help		
New (2)	Alt+Shift+N 🔸	🏄 Java Project	Sew	
Open File		🏫 Project	Select a wizard	
Close All	Ctrl+W Ctrl+Shift+W	🖶 Package		
	culuc	Class		
Save As	Ctri+S	G Enum	<u>W</u> izards:	
i Save All	Ctrl+Shift+S	Annotation	type filter text	
Revert		Source Folder	General	
Move		Folder		
Rename Refresh	F2 F5	S Untitled Text File	Project	
Convert Line Delimiters To	۶.	JUnit Test Case	Initial Text File     Adaptable Java Category	
👛 Print	Ctrl+P	💣 Task	Adaptable Class (4)	
Switch Workspace	•	📸 Example (3)	■ CVS ■ CVS ■ CVS	
🔁 Import		📑 Other	⊕ → Mylyn     ⊕→ XMI	
🛃 Export			Dether	
Properties	Alt+Enter		⊞- 🧀 Examples	
1 AdaptableClass.adpt [TestProject/	]			
2 conflict.xml [TestProject/] 3 conflict xsd [TestProject/]				
Evit				
			O < Back Next > Einish	Cancel

Fig.11: Generazione di una classe Adaptable dal menù generale.

🖨 Java - Eclipse Platform File Edit Source Refactor Navigate Search Run Window Help 🗂 • 🔚 🗁 💉 🔅 • 🕥 • 🗛 • 😕 Java Project e New 隚 Project... Select a wizard 💾 Package Explor 🙁 🍃 Hierarchy 🗖 🗖 瞪 Package (2) Selezionare New 🖃 😫 🖃 🞯 Class 🞯 Interface Wizards: 😭 Enum Alt+Shift+W 🕨 type filter text Show In Annotation 💕 Source Folder 🖃 🗁 General Copy 🗎 File 😂 Folder E Copy Qualified Name 📴 Folder 😭 File 💼 Paste Ctrl+V 👕 Project 📑 Untitled Text File 🗶 Delete Delete 😭 Untitled Text File 📑 JUnit Test Case 😑 🧀 Adaptable Java Catego Build Path . 👚 Task 1 Adaptable (4) 😐 🥪 CVS 🚵 Import... 📑 Example... (3) 😟 🗁 Java 🛃 Export... 😟 🗁 Mylyn 📑 Other... 😟 🧀 XML 🔗 Refresh F5 🗄 🧁 Other Alt+Enter 🛓 🧁 Examples Properties m Click con il tasto destro del mouse in un ? < Back Next > Finish Cancel punto gualsiasi del Package Explorer

Capitolo 3 - Sviluppo di un "Adaptable Java Project"

Fig.12: Generazione di una classe Adaptable da pop-up menù.

In tutti i casi suddetti, successivamente alla selezione dell'opzione "Adaptable Class", il wizard prosegue con la presentazione di una serie di dialog il cui compito è quello di consentire la scelta del progetto di appartenenza della nuova classe e di assegnarle un nome (il wizard ne propone uno di default) come illustrato in Fig.13.

(1) Il wizard propone un nome di default per la nuova classe Adaptable e presenta un tasto Browse per la selezione del progetto di appartenenza	(2) La pressione del tasto Browse attiva un dialog di selezione del progetto di riferimento (che deve essere necessariamente a tinologia Java)	
le New Adaptable Class	Una volta selezionato il progetto	
Adaptable Class Project must be specified.	premere OK	1
Project:	Select new Adaptable Class project	(3) Per terminare premere Finish
Adaptable Class name: AdaptableClass.adpt	Adaptable JavaProject	🗧 New Adaptable Class
		Adaptable Class         This wizard creates a new Adaptable Class with         *.adpt extension that can be opened by any         Project:       /AdaptableJavaProject         Browse         Adaptable Class name:       AdaptableClass.adpt
Enish Cancel		
	© OK Cancel	7 Einish Cancel

Fig.13: Indicazione dei riferimenti per una nuova classe Adaptable.

L'attivazione del "New Adaptable Class" wizard dalla menù bar e dal pop-up menù può essere resa ancora più immediata inserendo l'azione associata al "new Adaptable Class" direttamente tra le opzioni principali come in Fig.14. La richiesta avviene con la procedura riportata in Fig.15.

🖨 Java - Eclipse Platform					
File Edit Source Refact	File Edit Source Refactor Navigate Search Project Run Window Help				
📬 • 🖫 🗁   💣	🎄 • 🜔 • 💁 • 🛛 🍰	🖶 🞯 • 🕴 😂 🖉	👂 i 🖢 - 🖗 - 👳 🔶		
増 Package E 🛛 🏌	Hierarchy 🗖 🗖				
() 수 수 @					
	New	1	🖄 Java Project		
	Go Into		🎦 Project		
	Open in New Window Open Type Hierarchy Show In	F4 Alt+Shift+W	Adaptable Class     Package     Class		
	📄 Сору	Ctrl+C	🗊 Interface		
	E Copy Qualified Name		💕 Source Folder		
	💼 Paste	Ctrl+V	🕼 Enum		
	💢 Delete	Delete	🧼 Annotation		
	Unit Test Case				
	Source	Alt+Shift+S	Untitled Text File		
	Refactor	Alt+Shift+T I	Folder		
Import					
	🛃 Export		📸 Example		
	🖑 Refresh	F5	📑 Other		

Fig.14: L'azione "New Adaptable Class" tra le opzioni principali del pop-up menù.

😂 Java - Eclipse Platform		
File Edit Source Refactor Navigate Search	Project Run Window (1) Seleziondre Shi	ndow
Image: Source Refactor Navigate Search       Image: Source Navigate Search <th>Project Run Window (1) Selezionare Win New Window New Edbar Open Perspective (2) Selezionare Customize Perspective Save Perspective Close Perspective Close Perspective Close Perspective Close Perspective Close Perspective Close Perspective Close AI Perspective Close AI Perspective Close AI Perspective Close AI Perspective Accentorsiche is selezionato (1) il Submenu New nella scheda Shortcuts Preferences</th> <th>Adopted as a cascade items to the following submerus. The selections made will only affect the current perspective (Java). Select the shortcuts that you want to see added as cascade items to the following submerus. The selections made will only affect the current perspective (Java). Shortcut Categories: Adaptable Java Category (4) Inserine il segno di spunto in conrispondenzo di Adaptable Class (4) Inserine il segno di spunto in conrispondenzo di Adaptable Class (4) Inserine il segno di spunto in conrispondenzo di Adaptable Class (5) Shortcut Feam (6) Shortcut (7) Feam (7) Feam (7) Shortcut (7) Feam (7) Shortcut (7) Feam (7) Shortcut (7) Feam (7) Shortcut (7) Feam (7) Shor</th>	Project Run Window (1) Selezionare Win New Window New Edbar Open Perspective (2) Selezionare Customize Perspective Save Perspective Close Perspective Close Perspective Close Perspective Close Perspective Close Perspective Close Perspective Close AI Perspective Close AI Perspective Close AI Perspective Close AI Perspective Accentorsiche is selezionato (1) il Submenu New nella scheda Shortcuts Preferences	Adopted as a cascade items to the following submerus. The selections made will only affect the current perspective (Java). Select the shortcuts that you want to see added as cascade items to the following submerus. The selections made will only affect the current perspective (Java). Shortcut Categories: Adaptable Java Category (4) Inserine il segno di spunto in conrispondenzo di Adaptable Class (4) Inserine il segno di spunto in conrispondenzo di Adaptable Class (4) Inserine il segno di spunto in conrispondenzo di Adaptable Class (5) Shortcut Feam (6) Shortcut (7) Feam (7) Feam (7) Shortcut (7) Feam (7) Shortcut (7) Feam (7) Shortcut (7) Feam (7) Shortcut (7) Feam (7) Shor
	😰 Problems 🙁 🖉 Ø Javadoc 🙆 Declaratio	ОК         Салсе!

Fig.15: Procedura di richiesta "New Adaptable Class" tra le opzioni principali del popup menù.

La pressione del tasto Finish a conferma della creazione di una nuova classe Adaptable provoca le seguenti azioni:

- Generazione della nuova classe Adaptable con il nome e nel progetto di riferimento indicati nei vari dialog del wizard; La classe viene inizializzata sulla base di un template prestabilito e collocata in un source folder ("/adpt-src"), appositamente creato qualora non risulti già esistente;
- Generazione, se non esistente, di un file di default dei conflitti (conflict.xml) in una cartella prestabilita ("/adpt-conflict") anch'essa creata se non esistente. Il file viene inizializzato con un template predefinito;
- Generazione, se non esistente, di un file di default (conflict.xsd) contenente lo schema di definizione del file dei conflitti precedentemente menzionato nella stessa cartella del file dei conflitti ("/adpt-conflict"). Il file viene inizializzato con un template predefinito;
- Generazione, se non esistente e soltanto con lo scopo di organizzare i vari oggetti del progetto, di un folder destinato a contenere le Resources Definitions ("/adpt-resource").

#### 3.1. Adaptable Java Model decorators

La visualizzazione di tutti gli oggetti generati

in questa fase, che costituiscono il modello "Adaptable Java", è sempre associata a specifici "decorators" che ne rendono più immediata l'individuazione visiva in tutte le fasi di sviluppo e gestione del progetto.

La Fig.16 mostra i vari decorators visualizzati nella Package Explorer View che vengono riproposti in tutti i contesti operativi.



Fig.16: I decorators del modello Adaptable Java.

## 4. L'Adaptable Java Editor del ChameleonIDE

L'Editor per la gestione del linguaggio Adaptable Java, appositamente realizzato e fornito nel Plug-In ChameleonIDE, nasce con l'ambizione di "emulare" al meglio il JDT Editor di Eclipse per due motivi principali:

- L'indiscussa e universalmente riconosciuta completezza delle funzioni e degli strumenti offerti dal JDT Eclipse che lo hanno reso il principale ambiente IDE al mondo utilizzato nello sviluppo di progetti Java;
- Il tentativo di vincere la diffidenza comunemente riservata a tutti i nuovi tool di sviluppo, proponendo invece ai milioni di sviluppatori Java Eclipse un ambiente che si presenta immediatamente familiare e di semplice apprendimento.

Premesso che l'Editor viene associato automaticamente a tutte le classi Adaptable (estensione .adpt) ed è costruito sulla base dei costrutti e delle keywords del modello Adaptable Java, ne riassumiamo di seguito le funzionalità principali riportando nelle successive pagine alcune schermate esemplificative:

- Syntax Highlighting: la visualizzazione delle varie parti di codice secondo una colorazione completamente personalizzabile (commenti singoli, commenti multilinea, parole chiave, costanti, tipi, costrutti "speciali");
- Font Customization: delle varie parti di codice secondo font completamente personalizzabili nel tipo, nello stile e nella dimensione del carattere di scrittura;
- Auto-Indent Strategy: per l'applicazione automatica dei rientri di linea durante la scrittura del codice;
- Double-Click Strategy: per l'individuazione automatica di blocchi logici di codice quando si esegue un doppio click, ad esempio tra due parentesi graffe, tonde o quadre (aperte e chiuse) oppure tra una coppia di doppi apici;
- Text-Hovering: per la visualizzazione in modalità "sovrapposta ed evidenziata" della porzione di codice o dei token selezionati per un focus immediato degli stessi;
- Word Completion: attivabile come nel JDT con la sequenza di tasti Alt+"/" che scorre la lista delle keywords disponibili in modo da semplificare e velocizzare la scrittura dei programmi;
- Define Folding Region: che consente di nascondere momentaneamente blocchi di codice (visualizzando un segno "+" sulla Marker bar alla sinistra) liberando così il campo visivo al codice di interesse al momento. Il blocco nascosto può essere ripresentato in qualsiasi momento cliccando sul simbolo "+", che diverrà a quel punto un simbolo "-"; Il Folding Region si attiva

selezionando il blocco di codice da "nascondere", quindi premendo il tasto destro del mouse in un punto qualsiasi della Editor View, infine selezionando l'azione "Define Folding Region" dal pop-up menù attivato.

Content Assist: per la presentazione di templates completamente personalizzabili volti alla semplificazione del processo di scrittura del codice. I templates disponibili sono attivabili, come nel JDT, con la sequenza di tasti Ctrl+Space e si propongono automaticamente nel rispetto della porzione di codice eventualmente già digitata al momento della loro attivazione. Qualora il template selezionato presentasse delle variabili, esse potranno essere valorizzate una ad una mediante l'utilizzo del tasto di tabulazione, completandosi automaticamente negli altri punti in cui risultassero ripetute, così come avviene nella JDT. La gestione dei templates, particolarmente ricca di contenuti, viene descritta in un apposito paragrafo.

🖶 Java - AdaptableJavaProject/adpt-s	src/AdaptableClass.adpt - Eclipse Platform	
File Edit Source Refactor Navigate Sear	rrch Project Run Window Help	
i 🗈 • 🗟 🚔 💣 i 🕸 • 🔾 • 💊	・ :	
🛱 Package E 🕺 🍃 Hierarchy 🗖 🗖	MC *AdaptableClass.adpt ⊠	- 8
← ← ← ← ← ← ← ← ← ← ← ← ← ← ← ← ← ← ←	<pre>// This class can be adapted by the alternatives below Badaptable public class AdaptableClass (</pre>	^
rc ■ ■ JRE System Library [jre1.6.0_03] ■ ■ Referenced Libraries	<pre>adaptable void MethodName(); adaptable void MethodName1();</pre>	
dept-conflict     conflict.xml     X conflict.xsd		
adpt-resource	(3) Folding Kegion           @_lternative class AlternativeAdaptableClass ada AdaptableClass (	
(1) dage setting	alternative class AlternativeAdaptableClass adapts AdaptableClass (	
(1) decordiors	<pre>tag(TagName) public void MethodName1() (</pre>	
	•	
	(5) Text Hovering	~
		2

Fig.17: Syntax Hihglighting, Editor Font, Auto Indentation, Double-Click, Text Hovering e Folding.



Fig.18: Content Assist con Templates.

### 4.1. Annotation e Resources Annotation wizard

Particolare interesse riveste la gestione delle Resources Annotation che offre la possibilità di elencare in maniera completamente dinamica le caratteristiche dei dispositivi da indicare nella costruzione di una applicazione adattabile.

Il prodotto consente in fase di editing di attivare un vero e proprio wizard dedicato alla costruzione di una istruzione resourceAnnotation, parametrizzata sulla base delle dichiarazioni presenti nel file ResourceDefinition.xml desiderato, che può essere selezionato direttamente a runtime. La costruzione dinamica dei dialogs, di cui il ChameleonIDE è provvisto, è resa possibile da uno specifico processo di traduzione istantanea in tecnica SAX/DOM che parte da un Resource Definition file di tipo .xml.

Il wizard viene attivato mediante un click con il tasto destro del mouse nel punto in cui si desidera inserire l'istruzione resourceAnnotation, e attraverso opportuni passi guidati si completa con l'inserimento dell'istruzione parametrizzata oltre che l'inserimento dell'istruzione di import del package di riferimento della classe Annotation.java. Anche in questo caso è possibile indicare un nome package a proprio piacimento modificando quello preimpostato di default.

Una rappresentazione dell'intero processo è raffigurata nelle Fig.19, Fig.20 (esempi di dialog dinamici) e Fig.21 mentre due esempi di Resource Definition sono rappresentati in Fig.22.

ad	' This class can be laptable public cla	<i>adapted by the ai</i> ss AdaptableClass	ternatives below (		Find:
	adaptable void M	ethodName();	E Resource Definition file selection		
> ( - -	<ul> <li>Undo Typing Revert File Save <ul> <li>Save</li> <li>Site possibile modifice:</li></ul></li></ul>	Ctrl+Z are il package in cui tion class (di default allo del Chameleon el Ctrl+V	Annotation class location: chameleon.programmingmodel.Launch.Annotation		e o
1	Shift Right				
P 0 errc De:	Shift Left (2) Seleziona Insert Resource Annotation Run As Debug As Validate Team Compare With Replace With Preferences Content Assist Define Folding Region	Ctrl+Space	Control <ul> <li>Resources Definition File Selection</li> <li>Select an item to open (? = any character, * = any string)</li> <li>Presour*.xml</li> <li>Matching items:</li> <li>ResourcesDefinition.xml</li> <li>ResourcesDefinition.xml</li> <li>(5) Ung volta selezionato il file di interesse premere OK</li> </ul>	);	· ·
	Remove from Context Click con il tasto destro de qualsiasi della Package E	Ctrl+Alt+Shift+Down I mouse in un punto xplorer View	AdaptableJavaProject/adpt-resource	Canc	el

Fig.19: Insert Resource Annotation dialog.

Resource Definitio	n file selection	(6) A questo punto il dialog di sele
Resources <u>d</u> efinition file:	/AJavaProject/adpt-resource/ResourcesDefinition Browse	del tile risulta completato Procedendo con OK si attiva il dinamico per la selezione dei desiderati costruito a partire de Definition file indicato
Annotation class location:	chameleon.programmingmodel.Launch.Annotation	Il nome del package di riferim classe Annotation può essere a proprio piacimento
	OK Cancel	

#### (7) Alcuni esempi di dialog dinamico

Resource Annotation	Resource Annotation
✓ Time 60	🗹 Bluetooth 🛛 true 💌
🗹 Bluetooth 🛛 🗗 False 💌	Power 220
Energy	🗌 3DCaps 🛛 true 🔽
Eunctionalities	Wireless true 💌
ColorDisplay true V	false Battery
[false ☐ ScreenRes low ♥	OK Cancel
OK Cancel	(8) Il tasto OK conclude l'operazion

Fig.20: Alcuni dialog "dinamici" di Insert Resource Annotation.



Fig.21: Risultato della Insert Resource Annotation.

```
<?xml version="1.0" encoding="UTF-8"?>
<declaration>
 <enumeration id="res">
    <value id="low"/>
    <value id="medium" />
    <value id="high" />
 </enumeration>
    <resource id="Time" type="natural" summable="true" />
    <resource id="Bluetooth" type="boolean" summable="true" />
    <resource id="Energy" type="natural" summable="true" />
    <resource id="ScreenRes" type="res" summable="true" />
    <resource id="ColorDisplay" type="boolean" summable="true" />
    <resource id="Functionalities" type="res" summable="true" />
</declaration>
  <?xml version="1.0" encoding="UTF-8"?>
<declaration>
 <enumeration id="res">
    <value id="low" />
    <value id="medium" />
    <value id="high" />
 </enumeration>
 <resource id="Power" type="integer" summable="true" />
 <resource id="Bluetooth" type="boolean" summable="true" />
 <resource id="3DCaps" type="boolean" summable="false" />
 <resource id="Battery" type="integer" summable="true" />
 <resource id="Wireless" type="boolean" summable="false" />
</declaration>
```

Fig.22: Due esempi di Resource Definition.

E' inoltre possibile inserire velocemente anche altre istruzioni riferite alle Annotations (callAnnotation e loopAnnotation) avvalendosi dei templates opportunamente predisposti e richiamabili con la sequenza di tasti Ctrl+Space (Content Assist) come in Fig.23.



Fig.23: Content Assist con Templates per loopAnnotation e callAnnotation.

### 5. L'Adaptable Java Build

Una volta editato il nostro programma si passa al processo di Building.

Se non si è provveduto in precedenza è necessario a questo punto operare la procedura descritta nel precedente capitolo "Configurazione del Plug-In ChameleonIDE".

L'attivazione del Building avviene dal menù bar ed è collocata insieme alle altre funzioni di Build di Eclipse (Build All, Build Working Set, Build Automatically, ...) presenti sotto la voce Project.

In questo nuovo contesto compare la nuova opzione "Build Adaptable Class" che se richiesta attiva uno specifico dialog mirato alla composizione dei parametri di Building.

I parametri selezionabili riguardano:

La classe Adaptable da compilare;

- L'indicazione di compilazione di un'intero source folder (quindi di tutte le classi Adaptable in esso presenti). Se richiesta il source folder coinciderà con quello in cui è presente la classe Adaptable selezionata;
- Il Conflict Declaration file da utilizzare (se richiesto)
- Il ResourceDefinition file da utilizzare (se richiesto)
- L'indicazione di compilazione delle classi Java: Se non richiesto esplicitamente con questo parametro il processo di Building termina con la traduzione delle classi Adaptable in classi Java standard (.java), senza proseguire con la generazione dei file .class. Si è preferito infatti lasciare completa libertà di scelta all'utente che potrebbe avere l'esigenza di ultimare il processo in un secondo momento (ad esempio a fronte di qualche particolare caso in cui sia necessario modificare direttamente le classi Java standard) avvalendosi direttamente del Java Incremental Builder di Eclipse (JDT) oppure riattivando nuovamente il Chameleon Builder, questa volta richiedendo il processo completo;
- L'indicazione della directory in cui verranno generati i file .class (richiesto soltanto se in precedenza era stato indicato di eseguire completamente il processo di Building);

Al termine del processo di Building compare un dialog di conferma del refresh automatico del workspace e le classi generate appariranno nel folder di output indicato.

Una importante caratteristica del processo di Building è la sua attivazione mediante la tecnica delle Launch Configuration che consente di riproporre sempre gli ultimi processi eseguiti direttamente dalla menù bar, senza ripetere nuovamente l'intero processo di composizione dei parametri di Building.

Premendo infatti il tasto dalla toolbar si attiva automaticamente l'ultima configurazione "Java Or effettuata.

E' inoltre possibile mantenere una specifica Launch configuration di interesse richiamandola dal menù "Run As" (procedendo poi con "Open Run Dialog...") e rinominandola. In questo modo non verrà ricoperta da successive richieste di Building (che per default assumono sempre lo stesso nome "BuildAdptClass") e potrà quindi essere riutilizzata all'occorrenza (Fig.24).

🖶 Java - AdaptableJavaProject/adpt-src/AdaptableClass.ad	pt - Eclipse Plat	t 🗧 Run 🛛 🗙	
File Edit Source Refactor Navigate Search Project Run Windov	v Help D 🗁 🛷 🗄 🖢	Create, manage, and run configurations Run a Java application	
Implementation     Implementation     Implementation     Implementation       <	on.programmi <u>can be</u> adapt <b>b</b> s Ada thodN	Image: Second	Find:
Show In Alk+Shift+W Copy Ctrl+C Copy Qualified Name Paste Ctrl+V Copy Qualified Name Paste Ctrl+V Ctrl+V Ctrl+Alk+Shift del mouse in un punto d Path qualified della Path qualified della Path Alk+Shift+S Explorer View actor Alk+Shift+T	+Down	Solution	An outli
Import      Export      Export      Refresh F5      Close Project     Assign Working Sets      Conserver and the set of the s	ation E lication] odiNarr thoun	Image: Contract of a stress of a stre	FF ernat la cl
Rufr As Debug As Validate (3) Selezionare Open Run I Team Compare With Restore from Local History Properties Alt+Enter	ialog bodNam stitui -cs.si	ava Applet Ait-Smit+X, A ava Application Ait-Shift+X, J en Run Dialog ame adatta il metodo Adapt Methor uisce un'alternativa valida *** size() = 0 (7) Selezionando nuovamente lo freccia in basso del menò Run (1) Package E (2) I Heurard I BuildAdptClass1 adatt & Run As Open Run Dialog (2) Selezionando nuovamente lo freccia in basso del menò Run (1) Selezionando nuovamente lo freccia in basso del menò Run (2) Selezionando nuovamente reference del freccia in basso del menò Run (2) Selezionando nuovamente reference del freccia in basso del menò Run (2) Selezionando nuovamente del freccia in basso del menò Run (2) Selezionando nuovamente del freccia in basso del menò Run (2) Selezionando nuovamente del freccia in basso del menò Run (2) Selezionando nuovamente del frecia in basso del menò Run (2) Selezionando nuov	pt - I w He 29 C 20 C can Lic c
AdaptableJavaProject		si norera la nuova configurazione organizer avides successive compilazioni adaptable adaptable s	voi

Fig.24: Richiamo e ridenominazione di una Launch Configuration.

Le figure successive (Fig.25, Fig.26 e Fig.27) illustrano il processo di selezione dei parametri di Building, l'attivazione del processo ed il risultato ottenuto.

🚔 Java - AdaptableJavaProject/adpt	-sr <mark>c/Adaptab</mark> leClass.adpt - Eclipse Pl	atform
File Edit Source Refactor Navigate Sea	arch Project Run Window Help	
(1) Selezionare Project	Open Project Close Project	] - ∰ - <b>⇔ ↔ -</b> ↔ -
🛱 Package E 🛛 🍃 Hierarchy 🖵 🗖	Ruid All (2) Selezionare Build	Adaptable Class
(> -> @   🖻 🔩 💱 ▽	AC Build Adaptable Class	ingmodel.Launch.Annotation;
😑 📂 AdaptableJavaProject	Build Project	(3) Indicare i parametri
😕 src	Build Working Set 🕨	lantableClass { di Building nel dialog
JRE System Library [jre1.6.0_03]	Clean	
Referenced Libraries	Build Automatically	🖶 Build Adaptable Classes 🛛 🛛 🔀
conflict.xml	🔊 Generate Javadoc	
conflict.xsd	Properties	Adaptable Class:
🖻 🧑 adpt-resource	tag(TagName) mublic y	
ResourcesDefinition.xml	Annotation.resourceAnnot	Process all Adaptable Classes in the same folder
adot-src	)	Conflict declaration file: Browse
AC: AdaptableClass.adpt		
	}	Resources definition file: Browse
		Outout <.iava> Folder:
		Compile <.java> classes 🔘 yes
	<	
	Replans ? @ Javados & Dad	Output < bio> Folder
	Description	
		Compiler options:

Fig.25: Attivazione del processo di Building.

(4) Dialog di selezione della classe Adaptable		(6) Dialog di selezione del Resource Definition file
Adaptable Class Selection		Resources Definition File Selection
Select an item to open (? = any character, * = any string)	Build Adaptable Classes      Adaptable Class: //AdaptableJavaProject/adpt-src/AdaptableClass.ac/     Provess all Adaptable Classes in the same folder      Conflict declaration file: //AdaptableJavaProject/adpt-conflict/conflict.xml     Browse      Decourses definition file: //AdaptableJavaProject/adpt-conflict/conflict.xml	Select an item to open (? = any character, * = any string)
	Qutput <.java> Folder: /AdaptableJavaProject/arc Browse	OK Cancel
(5) Dialog di selezione del Conflict file	Cgmpile <,java> classes () yes ② no Ogtput <.bin> Folder: Browse	(7) Dialog di selezione della directory di output per i files java generati E Folder Selection Output <.java> folder selection
geocnifict.xml     Gooffict.xml     Gooffict.xml     Gooffict     OK Cancel	Compiler options:	<ul> <li>AdaptebilavaProject</li> <li>AdaptebilavaProject</li> <li>adpt-conflict</li> <li>adpt-source</li> <li>adpt-src</li> <li>bin</li> <li>met</li> </ul>
		⑦ OK Cancel

Fig.26: Selezione dei parametri di Building.

😂 Java - AdaptableJavaProject/src/A	.daptableClass_1.java - Eclipse Platform
File Edit Source Refactor Navigate Sea	arch Project Run Window Help
📬 • 🗒 🖆 💣   🎄 • 🔘 • 🍳	• * i 🖄 ₩ @ • i @ <mark>@ # i ]                                 </mark>
(8) La classe .java di output è stata	🕼 AdaptableClass.adpt 🚺 AdaptableClass_1.java 🛛
generata nel folder richiesto	import chameleon . programmingmodel . Launch . Annotation ;
🖃 😂 AdaptableJavaProject	
😑 🗁 src	public class AdaptableClass 1 (
(derault package)     (derault package)     (derault package)	
I A JRE System Library [jre1.6.0_03]	void MethodName ( )
Referenced Libraries	( Annotation , resourceAnnotation ( " Bluetooth(true), Energy(220)" ) ;
e conflict	· · · · · · · · · · · · · · · · · · ·
Conflict.xsd	
🖃 🧑 adpt-resource	
ResourcesDefinition.xml	(9) Il codice della classe .iava generata
adpt-src	
AdaptableClass.adpt	
	(10) Il report del processo di Building
	😰 Problems @ Javadoc 😥 Declaration 📮 Console 🕴 💿 📓 🕷 🙀
	<terminated>BuildAdptClass [Java Application] C:\Programmi\Java\jre1.6.0_03\bin\javaw.exe (27/giu/08 01:22:23)</terminated>
	Il metodo adaptable MethodName della classe adaptable AdaptableClass è stato definito nell'alternativa
	<pre>in metodo alternative methodwame della Classe alternative AlternativeAdaptableClass e Stato dichiarato *** ClassAdaptable AdaptableClass</pre>
	# classi Alternative 1
	# metodi Alternative 1
	# 15: 1 ************************************
	*** Il metodo Alter MethodName adatta il metodo Adapt MethodName
	*** L'Insieme Stabile costituisce un'alternativa valida ***
	///:::/lass #daptable=CS.Size()= U
< >	
<u></u>	

Fig.27: Risultato del processo di Building.

## 6. Adaptable Java Preferences Pages

Il Plug-In ChameleonIDE offre un'ampio margine di personalizzazione dell'ambiente di sviluppo attraverso le Preferences Pages, già familiari perché di comune utilizzo sia per le configurazioni di Workbench, sia per quelle del Java Development Environment:

- Editor Font: consente la scelta del tipo di carattere da utilizzare nell'Adaptable Java Editor, nonché delle sue dimensioni e dello stile (normale, grassetto, corsivo);
- Syntax Coloring: offre la possibilità di personalizzare i colori di varie porzioni di codice quali
  - Single-line comments;
  - Multi-line comments;
  - Keywords: sia quelle Java standard (p.es. class, public, private, ...), sia
     le Adaptable Java (p.es. adaptable, adapts, alternative, ...);
  - Types: ossia i tipi Java (p.es. boolean, void, char, ...);
  - Costants: le costanti true, false e null;
  - Specials: ossia le istruzioni Annotation.resourceAnnotation, Annotation.callAnnotation e Annotation.loopAnnotation;
- Templates: Per la definizione dei templates utilizzati dal Content Assist durante la fase di editing. Il Plug-In ChameleonIDE nasce già corredato da una serie di templates di base che può essere ulteriormente arricchita a proprio piacimento e senza limiti da templates personalizzati oppure da tutti o parte dei templates già presenti in contesti compatibili con il contesto Adaptable Java (context=java e context=javadoc) mediante funzioni automatiche di import.

Le Adaptable Java Preferences Pages vengono attivate con la procedura descritta nella figura seguente (Fig.28):



Fig.28: Attivazione delle Adaptable Java Preferences Pages.

Ciascuna delle nuove Preferences Pages consente sempre di ripristinare i valori di default attraverso un apposito comando a bottone (Restore Defaults). Le figure successive rappresentano il contesto operativo delle Preferences Pages appena descritto.



Fig.29: Preferences Page dell'Editor Font ed effetto delle variazioni operate.

😂 Java - AdaptableJavaProject/adpt	t-src/AdaptableClass.adpt - Eclipse Platform			- 2 🛛
File Edit Source Refactor Navigate Se	sarch Project Run Window Help	(a)		
i 🗈 • 🗟 👜 💣 i 🏇 • 🔘 • 🍳	🎍 • 🗄 😫 🞯 • 🛯 🕭 🖨 🖉 🗄 • 🖓 • *	★> <> → Preferences		💶 🚨 🔛 🔡 Java
😫 Package E 🕺 🍃 Hierarchy 🖓 🗖	🚺 🕺 AdaptableClass.adpt 🗴 🚺 AdaptableClass_1.jav	(1) Selezionare la pagir va Syntax Coloring	Syntax Coloring	🗘 - 🗘 - 🗖 Task List 🕅 🗖 🗖
← ← ← ← ← ← ← ← ← ← ← ← ← ← ← ← ← ← ←	<pre>import chameleon.programmingmode. // This class can be adapted by : adaptable public class Adaptable</pre>	Launch. An Bread General - Adaptable Java Editor The alterna - Adaptable Java Editor Tass (	Customize color for Svntax Highlig (2) Operare Multi-line comment:	Jhtina. ≥ le scette desiderate ■ ØBold ♥ Italic d: ► All ►
Gefault package)	adaptable void MethodName();	- Templates - Ant - Cache	Single-line comment:	Bold VItalic
Interface System Library [jre1.6.0_03]     Interface System Libraries     Interface System Libraries	)	i ⊡ Install/Update	Keyword:	Bold Italic
conflict.xml	alternative class AlternativeAdapt	tableClass Mylyn Run/Debug	Java Constant:	Bold Italic
digitaria di la construcción de la construcció	tag(TagName) public void Metho Annotation.resourceAnnotation(" )	Luctooth(tValidation → Web and XML	Special:	
adpt-src	) (5) L'effetto delle scette nell'Editor	Colore ?X	(4) Pro re	emere Apply per indere attive le scette
		Colori di base:	Restore	Defaults Apply
	3			
	Problems @ Javadoc 😥 Declaration 🚍 Console			= × 🔆 🖬 🖬 🖅 🖤 📑 🗉 - 🗂 - 🗖
	<pre><terminated>BuildAdptClass[Java Application] C:\Programminated&gt;BuildAdptClass[Java Application] C:\Programminate Il metodo adaptable MethodName della Il metodo alternative MethodName del</terminated></pre>		D1:22:23) lass è stato definito nell' ativeAdaptableClass è stato	alternativa AlternativeAdaptableClas 📥 dichiarato nella classe adaptable A
	<pre>*** ClassAdaptable AdaptableClass # classi Alternative 1 # metodi Alternative 1 # IS: 1</pre>	Colori personalizzati:	(3) La pressione del tasto colorato attiva la visualizzazione della	
	**************************************	t Definisci colori personalizzati >>	tavolozza dei colori	_
	???!!!Class Adaptable-cs.size()=			
< >				
÷ 0*			Writable Insert 14:2	
🛃 start 👘 🏉 🚱 🚱 📄 Plug	g-in Development 🐻 ChameleonIDE User 🐻	Attività - Microsoft W 🖉 🚑 Java - Adapta	ableJav 🦉 Immagine - Paint	(2) 10

Fig.30: Preferences Page del Syntax Coloring ed effetto delle variazioni operate.

Preferences			
type filter text	Adaptable Java Templates I tasti New e Edit consentono la definizione o la modifica dei tem	nplates (1)	⇔ → →
🖽 General	Il tasto Remove provoca la cancellazione del template selezionato al ma	mento	<u>N</u> ew
Syntax Coloring	adaptable_class Adaptable J adaptable class template	on	
Templates	adaptable_met Adaptable J adaptable method template	on	
🗄 - Ant	alternative Adaptable J alternative template	on	Remove
Cache	callAnnotation Adaptable J callAnnotation template	on	
⊞ Help	callAnnotation Adaptable J callAnnotation template	on	
	loopAnnotation Adaptable J loopAnnotation template	on	Restore Removed
instan,opuato ⊞-lava	loopAnnotation Adaptable J loopAnnotation template	on	Devent to Defeult
i sara ⊕-Mvlvn	resourceAnnot Adaptable J resourceAnnotation template	on	Revert to Derault
<ul> <li>Run/Debug</li> <li>Team</li> <li>Validation</li> <li>Web and XML</li> </ul>	(2) Il preview visualizza in anteprima il template selezionato al momento         Preview:       I tasti         // This class can be adapted by the alternatives below       ar	i Import ed Itivano il pro	Import Export Export (3) cesso di
	adaptable public class \${classname} {     adaptable \${returntype} \${methodname}();     }     alternative class \${alternativename} adapts \${classname} {         Let the thouse adapts \${classname} {         Let the thouse adapts \${classname} } {         Let the thouse adapts \${classname} } {     }	ione o espo dei te Restore <u>D</u> efau	Its Apply
0		ОК	Cancel

Fig.31: Preferences Page dei Templates (funzioni principale).

Preferences		
type filter text	Adaptable Java Templates	⇔ - ⇔
General Adaptable Java Editor Syntax Coloring Templates Coloring Edit Adaptable Java	Name       Context       Description       Itasti New ed Edit (1)         Image: Structure of the struct	<u>N</u> ew <u>E</u> dit <u>R</u> emove
Image: Name:         Jadaptable_class           Image: Description:         adaptable class	i Context: Adaptable Java Templates V Automatically insert on on on template templat	Restore Removed Revert to Default
<ul> <li>Pattern: // This class can adaptable public adaptable \${r</li> <li>}</li> <li>alternative class</li> </ul>	be adapted by the alternatives below class \${classname} { eturntype} \${methodname}(); \${alternativename} adapts \${classname} {	Import
(2) La pressione propone un selezione di u	del tasto Insert Variable nenù a tendina per la ariabili predefinite OK Cancel Restore Defe	aults Apply
0	ОК	Cancel

Fig.32: Preferences Page dei Templates (funzioni di editing).



Fig.33: Preferences Page dei Templates (funzione di import).



Fig.34: Preferences Page dei Templates (funzione di export).

## 7. Appendice A - Adaptable Java Icons

AC	Classe Adaptable		
	Adaptable Java class source folder		
	Resource Definition file o folder		
0	Conflict declaration file o folder		
*	Toolbar Action per New Adaptable Class		
	Adaptable Java Template (Content Assist)		
AC	New Adaptable Class Wizard		
?	Link alla pagina web dei parametri di compilazione		

### 8. Appendice B - Le classi Launch. java e Annotation. java

La classe Launch.java

```
package chameleon.programmingmodel.Launch;
import chameleon.programmingmodel.Tool.*;
 * Title: Launch 
  Description: Questa classe viene usata come esecuzione del
                   Programming Model per la fase di Build.
                   Parametri ricevuti:
                   - Path completo della Adaptable Class da compilare
                     (se il Path termina con "/" si intende una directory);
                   - Path completo del Conflict file
                     (se non previsto viene indicato con "none");
                   - Path completo del Resources Definition file
                     (se non previsto viene indicato con "none");
                   - Path completo indicante la directory di output
                    dei file .java prodotti terminante per "/";
                   - Path completo indicante la directory di output
                    dei file .bin prodotti terminante per "/"
                     (se non previsto va indicato con "none");
                   - opzioni di compilazione
                     (se non previsto va indicato con "none").
 * 
 * Copyright: Copyright (c) 2008
 * @author Dipartimento di Informatica UNIVAQ
 * @version 1.0
 */
public class Launch {
  /**
   *
    * Metodo usato per lanciare il tool
   * viene istanziato un oggetto di tipo
   * MainProgModel e quest'ultimo richiama
   * il metodo MainUnit
   * @param args[0]
                        [String] : file o directory delle adaptable classes
                                   (se directory termina con "/")
   * @param args[1]
                       [String] : file dei conflitti | none
   * @param args[2]
                       [String] : file delle resources definition | none
                       [String] : directory di output (.java)
    * @param args[3]
                                   (termina con "/")
    *
                       [String] : directory di output (.bin) | none
     @param args[4]
                                   (se indicata termina con "/")
```

```
* @param args[5..n] [String] : riga delle opzioni di compilazione | none
*/
public static void main(String[] args) throws Exception {
   System.out.println("Chameleon Programming Model Build Start ...");
    int numargs=args.length;
    System.out.println("Number of arguments provided ..... " + numargs);
    if (numargs==0) {
          System.out.println("Chameleon Programming Model Build End .....: No argument found");
          return;
    }
   int i = 0;
   while (i < args.length) {</pre>
          System.out.println("args["+i+"]: "+args[i]);
          i++:
   }
    MainProgModel prebuild = new MainProgModel();
    prebuild.MainUnit(args);
    }
}
```

#### La classe Annotation.java

package chameleon.programmingmodel.Launch;

```
public class Annotation {
       public static void resourceAnnotation(String annotation) {
              // annotation deve essere una lista separata da virgole di ResID(Valore)
       }
       public static void loopAnnotation(boolean monitored, int occurrences) {
               // se monitored è vero vuol dire che il programmatore non è certo che il
               // ciclo verrà ripetuto al max occurrences volte, quindi il numero di volte
               // viene gestito come una variabile.
              // Le variabili appartenenti a cicli monitored vengono stampate in un file
              // LoopAnnotation.xml
               // ognuna associata con il rispettivo valore di occurrences
               // se monitored=false si assume che il ciclo viene ripetuto
              // occurrences volte.
       }
       public static void loopAnnotation(int occurrences) {
              // equivale a chiamare loopAnnotation(false, occurrences);
       }
       public static void callAnnotation(boolean monitored, int occurrences) {
               // messa prima di una chiamata a funzione (mutualmente) ricorsiva,
               // dice quante volte la chiamata deve essere ripetuta
              // monitored: come per le loopAnnotation
       }
       public static void callAnnotation(int occurrences) {
              // equivale a chiamare callAnnotation(false, occurrences);
       }
```

}