ЛАБОРАТОРНА РОБОТА №1. ПІДГОТОВКА ПРОГРАМНОГО ПРОЕКТУ

Тема: Підготовка програмного проекту

Мета: Отримання базових навичок з використання мови XML. Вивчення структури типового програмного проекту, форматів стандартних файлів опису проекту. Вивчення формату JAR. Здобуття навичок з використання засобів автоматизації процесу збірки програмних проектів на мові Java - Apache ANT (Another Neat Tool). Розробка програмного проекту на основі типового прикладу.

Довідка

Розширювана мова розмітки (англ. *Extensible Markup Language*, скорочено **XML**) — запропонований консорціумом World Wide Web (W3C) стандарт побудови мов розмітки ієрархічно структурованих даних для збереження та обміну. Є спрощеною підмножиною мови розмітки SGML. XML документ складається із текстових знаків, і придатний до читання людиною.

Логічна структура. Інструментом розмітки в XML є тег, який використовується для визначення меж елемента. Тег буває: початковий (<Ім'я-елемента>), кінцевий (</Ім'я-елемента>) та порожній або закритий (<Ім'я-елемента/>). XML документ має ієрархічну структуру, яка складається з елементів, асоційованих з ними атрибутів та інструкцій. Непорожній елемент визначається парою тегів (початковий і кінцевий) з ім'ям цього елементу та тілом, що міститься між цими тегами. Тіло елементу складається з інших елементів та/або тексту. Порожній (без тіла) елемент може визначатися за допомогою одного порожнього тегу з ім'ям цього елементу. Атрибути є послідовністю пар ключ-значення (назва

атрибута="значення атрибута") і знаходяться або у початковому, або у порожньому тезі і асоціюються з елементом, ім'я якого зазначено в тезі. Також, за допомогою спеціальних тегів визначаються інструкції обробки документу (<?Обробник параметр ?>) та коментарі (<!-- Текст коментаря -->).

Коректність. Коректний документ (well-formed document) відповідає всім синтаксичним правилам XML. Документ, що не є коректним, не може називатись XML-документом. Коректний XML документ має відповідати :

- Документ має лише один елемент в корені.
- Непорожні елементи розмічено початковим та кінцевим тегами. Порожні елементи можуть помічатись «закритим» тегом.
- Один елемент не може мати декілька атрибутів з однаковим іменем. Значення атрибутів знаходяться або в одинарних ('), або у подвійних (") лапках.
- Теги можуть бути вкладені, але, не можуть перекриватись. Кожен некореневий елемент мусить повністю знаходитись в іншому елементі.
- Фактичне та задекларованe кодування документа мають збігатись.

Валідність. Документ називається валідним (англ. *valid*), якщо він є коректним та семантично вірним, тобто відповідає певному словнику XML, що визначається схемою (DTD, XML Schema або іншою).

JAR-файл — Java архів, що являє собою архів формату ZIP з додатковими метаданими в файлі маніфесту (META-INF/MANIFEST.MF). Маніфест може містити інформацію при назву проекту, версію, автора, стартовий клас, підпис файлів, тощо. Для створення JAR може бути використаний будь-який ZIP-сумісний архіватор. Найчастіше застосовують

спеціалізовані засоби для роботи з JAR — завдання <jar> для Ant або утиліту jar із JDK.

Apache Ant (англ. *ant* — мураха і водночас акронім — «Another Neat Tool») — java-утиліта для автоматизації процесу збирання програмного продукту. На відміну від make, утиліта Ant повністю незалежна від платформи, потрібна лише наявність на застосовуваній системі встановленої робочого середовища Java — JRE. Відмова від використання команд операційної системи і формат XML забезпечують переносимість сценаріїв.

Управління процесом збирання відбувається за допомогою XML-сценарію, який також називають Build-файлом (build.xml). Цей файл містить кореневий елемент- проект, що складається з елементів-цілей (<target>). Цілі є еквівалентом процедур в мовах програмування і містять виклики команд-завдань (task). Завдання являє собою XML-елемент, що пов'язаний з java-класом, який виконує певну елементарну дію. Часто вживані завдання: javac, copy, delete, mkdir, jar. Між цілями можуть бути визначені залежності (атрибут depends) — кожна ціль виконується тільки після того, як виконані всі цілі, від яких вона залежить (якщо вони вже були виконані раніше, повторного виконання не здійснюється). Типовими прикладами цілей є clean (видалення проміжних файлів), compile (компіляція всіх класів), deploy (розгортання програми на сервері). Конкретний набір цілей та їхнього взаємозв'язку залежать від специфіки проекту. Ant дозволяє визначати власні типи завдань шляхом створення Java-класів, що реалізують певні інтерфейси, та зв'язування цього класу з елементом сценарію за допомогою елемента <taskdef>.

Завдання

1. Вивчити синтаксис та структуру мови XML. Вільно володіти поняттями тег, елемент, атрибут. Вміти редагувати XML-файли у текстовому редакторі.
2. Ознайомитись з призначенням і структурою архівів JAR. Вміти формувати архіви JAR та запускати на виконання Java-класи з архіву JAR за допомогою засобів командної строки. Знати призначення підпису архіву JAR.
3. Ознайомитись з засобом автоматизації збірки проектів ANT. Вивчити призначення і структуру файлу build.xml, його структурні компоненти – цілі, завдання, залежності, тощо.
4. З сайту лабораторії завантажити архів типового проекту для середовища Eclipse (template.zip). Ознайомитися з структурою каталогів типового проекту, XML-файлами опису проекту (.project, .classpath, build.xml).
5. Імпортувати типовий проект до робочого простору (workspace) середовища Eclipse. В пакеті com.lab111 запустити на виконання (run) клас TestMain. Ознайомитися з засобами використання ANT у середовищі Eclipse - редагування файлу build.xml, виконання цілей у відображенні (View) ANT. Виконати цілі ANT з середовища Eclipse та з командного рядка.
6. Модифікувати типовий проект для його використання у наступних лабораторних роботах. Обов'язково замінити назву і автора проекту в файлах опису проекту (.project, build.xml). Модифікувати файл build.xml таким чином, щоб у ньому були всі потрібні цілі, вільно володіти призначенням кожної цілі.

7. Розробити та перевірити в eclipse нову ціль в файлі build.xml згідно варіанту.

Варіанти завдання

Номер варіанту завдання обчислюється як залишок від ділення номеру залікової книжки на 9.

0. Створити каталог new-out. Скопіювати в цей каталог всі файли проекту з розширенням jar.
1. Видалити з проекту всі файли з розширеннями tmp,jar,class крім тих, що починаються з літери "a".
2. Створити в каталозі out jar-архів з усіх файлів проекту з розширеннями java,js,html,htm. Скопіювати архів в корінь проекту.
3. Створити в каталозі out jar-архів з усіх файлів проекту з розширенням txt. Видалити такі файли з проекту.
4. Створити каталог build2. Виконати компіляцію сирцевих файлі проекту в створений каталог.
5. Видалити з каталогу out і нижче всі файли, що мають розширення jar. Упакувати в jar всі файли каталогу src, що починаються з літери "z".
6. Створити jar-архів проекту з ім'ям, що містить дату і час створення. При створенні архіву оминати файли з розширеннями zip та jar.
7. Видалити з кореня проекту всі файли з розширенням jar. Упакувати в jar проект увесь проект окрім того, що міститься в каталозі out.
8. Виконати компіляцію тестової гілки проекту. Скомпільовані класи запакувати в jar.

Питання для самостійної перевірки

1. Призначення мови XML. Поняття словника.
2. Структура XML-документу. Види тегів.
3. Правильність XML-документу. Умови для well-formed XML-документу.
4. Структура JAR-архіву, призначення маніфесту.
5. Як створити JAR-архів, який може бути виконаний. Як запускати на виконання класи в JAR-архіві.
6. Призначення каталогів в типовому проекті.
7. Призначення файлів .project, .classpath, build.xml в типовому проекті.
8. Призначення ANT. Його відмінність від make.
9. Структура файлу build.xml. Словник XML для ANT.
10. Цілі і задачі. Алгоритм створення цілей і задач.
11. Послідовність дій ANT після запуску з командного рядка. Синтаксис запуску ANT з командного рядка.
12. Засоби ANT для роботи з файловою системою (створення/видалення каталогів, копіювання тощо).
13. Засоби ANT для компіляції сирцевих файлів Java.
14. Засоби ANT для створення JAR-архіву.
15. Засоби середовища eclipse для роботи з ANT.

Протокол

Протокол має містити титульну сторінку, завдання, роздруківку змісту каталогу проекту з відповідними коментарями та роздруківку файлів .project, .classpath, build.xml з відповідними коментарями

Список рекомендованих інформаційних джерел

XML

- XML. Материал из Википедии – свободной энциклопедии. – http://ru.wikipedia.org/wiki/XML.
- Язык XML– практическое введение. – http://www.citforum.ru/internet/xml/index.shtml.

JAR

- Trail: JAR Files. – http://khpi–iip.mipk.kharkiv.edu/library/extent/prog/jar/index.html.

ANT

- Apache Ant. –http://ru.wikipedia.org/wiki/Apache_Ant.
- Ant за 10 шагов. –http://www.opennet.ru/base/dev/ant_10.txt.html.

ЛАБОРАТОРНА РОБОТА №2. ГРАФІЧНА НОТАЦІЯ UML, ДОКУМЕНТУВАННЯ ПРОЕКТУ

Мета: Ознайомлення з видами діаграм UML. Отримання базових навичок з використання діаграми класів мови UML. Здобуття навичок з використання засобів автоматизації UML-моделювання на прикладі ArgoUML/Umbrello. Документування проекту за допомогою JavaDoc.

Довідка

UML (англ. Unified Modeling Language) — уніфікована мова моделювання, використовується у парадигмі об'єктно-орієнтованого