



Міністерство освіти і науки України
Сумський державний університет

Методичні вказівки
до виконання лабораторних робіт з курсу
«Візуальні мови програмування»
для студентів напрямку підготовки 171 «Електроніка»
денної заочної та дистанційної форми навчання

Суми
Сумський державний університет
2017

Методичні вказівки до лабораторних робіт з курсу «Візуальні мови програмування» / Укладачі С.І. Проценко, М.Г. Демиденко.
– Суми: Сумський державний університет, 2017. – 51 с.

Кафедра електроніки, загальної та прикладної фізики

ПЕРЕДМОВА

Курс «Візуальні мови програмування» включає в себе два основні розділи, що направлені на вивчення таких технологій, як Flash та Flex. Перша – це система для створення векторної анімації, що має великі можливості програмування і на сьогодні використовується для інтерактивних додатків, як інтерфейс користувача в складних веб-орієнтованих системах. Flash від Macromedia являє собою симбіоз графіки, анімації та програмування. Ця технологія дозволяє створювати не тільки інтерактивні фільми, а навіть сайти і використовувати ефекти, які важко виконати за допомогою засобів HTML та JavaScript. Мовою технології Flash являється ActionScript – скрипкова мова, що буквально означає «мова дій».

У другій частині курсу вивчається технологія Flex, мовою якої являється не тільки ActionScript, а й MXML («мова розмітки»), що дозволяє створювати складні додатки з чіткою та зрозумілою структурою. Adobe Flex являється новою технологією для створення функціональних веб-додатків та традиційних програм, що відтворюються за допомогою Flash Player. Flex – це мова програмування, в основі якої лежить MXML, що в свою чергу заснована на розширеній мові розмітки (Extensible Markup Language, XML). Її використання робить створення додатків легким та ефективним. Flex - технологія поєднує всі переваги сучасних мов програмування та стандарти і методи веб-розробок.

Як і інтегроване середовище розробки Flash, Flex створює додатки, що відтворюються за допомогою Flash Player. Єдиною загальною рисою цих технологій є використання мови сценаріїв, а в іншому вони відмінні. Flash – це, в першу чергу, інструмент створення анімації та графічний редактор. Flex – засіб для побудови складних додатків.

У процесі виконання лабораторних робіт Ви ознайомитесь з основними можливостями цих технологій.

Лабораторна робота 1

Реакція на кнопку у вигляді анімації об'єкту. Об'єднання анімацій із різних файлів

Мета роботи: закріпити навички створення анімації форми та кольору об'єкту. Навчитися використовувати стандартні кнопки із бібліотеки і задавати для них події, а також об'єднувати анімації із різних файлів.

Елементи теорії. Анімувати об'єкт – означає змусити його плавно змінювати свої властивості. Стан об'єкту характеризується його розміром, кольором, положенням у просторі, формою. При створенні простої анімації, інструментами Flash змінюються ці властивості. З точки зору методу побудови, розрізняють два типи анімації: анімація форми (shape tweening) та анімація руху (motion tweening).

Анімація діє у відповідності з її описом на шкалі часу, причому кадри анімації слідують у послідовності один за одним. Для того, щоб перейти на довільний кадр (з якого далі продовжиться анімація) використовують команди управління.

ПОРЯДОК ВИКОНАННЯ РОБОТИ

1. На поточному шарі створити послідовність дій, що буде запускатися при натисканні на кнопку (наприклад, анімація переміщення об'єкту зі зміною форми та кольору);
2. Для створеної послідовності виконати наступні дії: виділити перший ключовий кадр і дати йому ім'я (ввести його у полі «Frame» панелі «Properties»);
3. Виділити останній ключовий кадр та відкрити вікно «Actions»;
4. У підгрупі «Movie Control» вибрати команду зупинки анімації (див. рис. 1);

Оскільки, послідовність дій не повинна виконуватися раніше ніж буде натиснута кнопка, на початковому кадрі необхідно поставити команду «stop()». Для цього, переміщують послідовність і звільненому кадру приписують

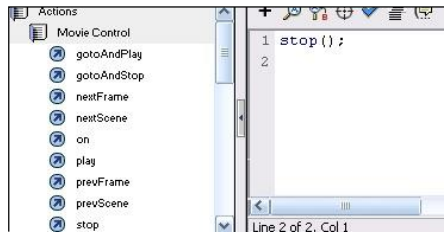


Рисунок 1 – Створення команди зупинки анімації

команду (див. рис. 2).

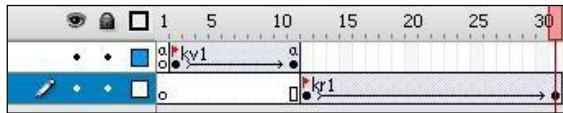


Рисунок 2 – Вигляд часової шкали після додавання команд

5. Створіть шар для кнопки; 6. Виберіть кнопку із стандартної бібліотеки: «Window → Common Libraries → Buttons»;

7. Припишіть їй події. Для цього необхідно: виділити кнопку, відкрити панель «Actions» і вибрати з підгрупи «Movie Control» команду «on» (встановити для неї значення «release») (див. рис. 3);

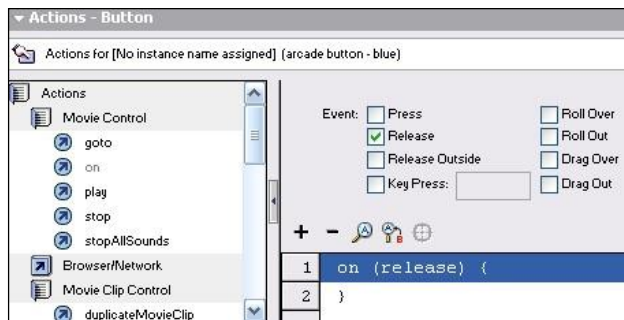


Рисунок 3 – Встановлення команди «on» для кнопки

8. Далі: виберіть команду «gotoAndPlay», у круглих дужках вкажіть ім'я кадру (у лапках), що буде програватися при натисканні на кнопку (в даному випадку ім'я першого ключового кадру) (див. рис. 4);

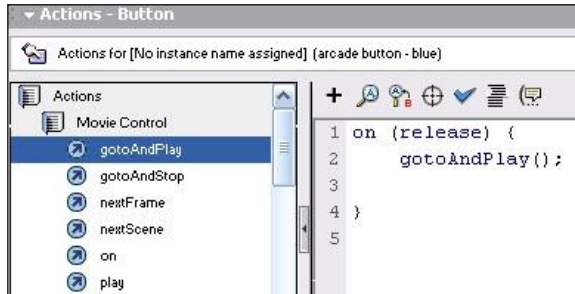


Рисунок 4 – Створення команди «gotoAndPlay(“Ім'я”)»

Для того, щоб об'єднати анімації з різних файлів необхідно виконати наступну послідовність дій:

9. Підготувати два файли з розширенням «.swf» (опубліковані), що містять різні анімації;

10. Відкрити файл «1.fl» і виділити ключовий кадр, після якого необхідно запустити анімацію з другого файлу (це може бути, наприклад, останній ключовий кадр послідовності);

11. Відкрити панель «Actions» і у групі команд «Browser/Network» вибрати «loadMovie». В області параметрів вказати ім'я файлу, що буде запускатися (з розширенням «.swf»), як показано на рис. 5:

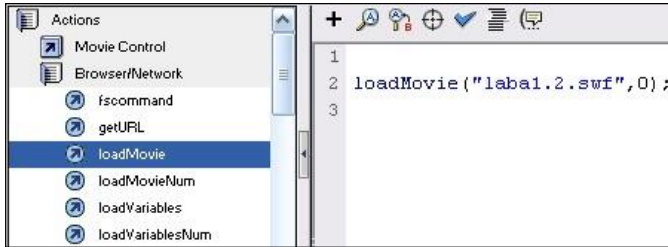


Рисунок 5 – Об'єднання анімацій з різних файлів

Таким чином, при натисканні на кнопку буде виконуватися анімація з файлу «l.swf», після чого автоматично завантажиться другий файл.

ЗМІСТ ЗВІТУ

1. Номер, тема та мета роботи.
2. Елементи теорії.
3. Програмний код для створених об'єктів.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Основи ООП у середовищі Flash. 2. Технологія Flash. Основні напрямки використання.
3. Типи файлів, які використовуються у середовищі розробки Flash. Опубліковані файли анімації.
4. Ключовий кадр. Використання кадрів для побудови анімації.
5. Описати алгоритм об'єднання анімацій із різних файлів.

ЛІТЕРАТУРА

1. Adobe flash CS3 professional. User guide / – California, USA: Adobe Systems Incorporated, 2007. – 277 p.
2. ActionScript 2.0. Справочник разработчика / Под ред. С.Н. Тригуб. – СПб.: Вильямс, 2005. – 894 с.
3. К. Мук. ActionScript для Flash MX. Подробное руководство. – СПб.: Символ, 2004. – 1120 с.
4. T. Green, D. Stiller. Foundation Flash CS3 for Designers. – California, USA: Friedsoft, 2007. – 556 p.

Лабораторна робота 2

Створення кнопки. Рух по заданій траєкторії

Мета роботи: навчитися створювати власні кнопки і описувати їх роботу у різних станах, а також задавати довільну траєкторію, за якою буде рухатися анімація.


Елементи теорії. Для переміщення об'єкту не по прямій, а за довільною кривою, необхідно створювати окремий шар, де ця траєкторія буде прорисована. Для цього треба натиснути кнопку  у лівому нижньому куті монтажного столу, або викликати контекстне меню для шару з об'єктом і вибрати пункт «Add Motion Guide» (добавити напрямну руху). У результаті виконання цих дій з'явиться новий шар:



Рисунок 1 – Окремий шар для траєкторії руху

На цьому шарі прорисовується траєкторія, що повинна починатися у центрі об'єкту на початковому ключовому кадрі і закінчуватися у центрі об'єкту на останньому ключовому кадрі. Для полегшення цих дій використовуються напрямні лінії. При чому, бажано відмінити режими притягнення до напрямних та об'єктів: «View → Snapping → Snap to Guides» (убрати мітку); і встановити режим притягнення кінців траєкторії до середини об'єкту, для цього на початковому ключовому кадрі на панелі «Properties» поставити мітку «Snap» (див. рис. 2).

Для створення лінії можуть використовуватися різні інструменти «Pencil Tool», «Brush Tool», «Line Tool».

ПОРЯДОК ВИКОНАННЯ РОБОТИ

1. Прорисуйте траєкторію у кадрі, де починається рух. Перейдіть на останній кадр і виберіть з контекстного меню



Рисунок 2 – Встановлення прив'язки кінців траєкторії до центру об'єкту

команду «Insert Frame». Загальний вигляд монтажного столу повинен бути таким:



Рисунок 3 – Вигляд монтажного столу після створення траєкторії

Як анімація може використовуватися об'єкт, що рухається за заданою траєкторією. В цьому випадку необхідно створити три окремі шари («Guide Layer» повинен бути невидимим).

1. Кнопку, що керує стартом анімації створить самостійно (на окремому шарі). Для цього необхідно вибрати «Insert → New Symbol», або натиснути Ctrl+F8. У вікні, що з'явиться вибрати пункт «Button» і ввести ім'я символу (див. рис. 4):

2. Вибрати даний символ із бібліотеки «Window → Library» і перетягнути в кадр. Відкриється монтажний стіл, що містить чотири кадри: «Up», «Over», «Down», «Hit», які відповідають різним станам кнопки (див. рис. 5):

Up – звичайний стан кнопки;

Over – курсор миші знаходиться над кнопкою;

Down – курсор знаходиться над кнопкою і натиснута клавіша миші;

Hit – звичайний стан кнопки, що містить посилання, яке користувач вже відвідував.



Рисунок 4 – Створення нового символу – кнопки

3. У кожному кадрі внесіть зміни та натисніть синю стрілку (вихід з режиму редагування).

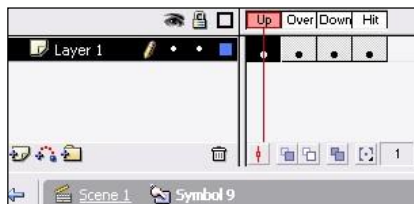


Рисунок 5 – Редагування станів кнопки

Таким чином, після натискання створеної кнопки, об'єкт починає рухатися по заданій траєкторій. Приблизний вигляд монтажного столу в цьому випадку:

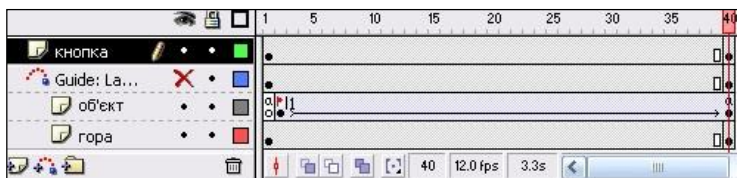


Рисунок 6 – Загальний вигляд монтажного столу

ЗМІСТ ЗВІТУ

1. Номер, тема та мета роботи.
2. Елементи теорії.
3. Загальний вигляд монтажного столу.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Створення кнопки. Основні кадри для елемента «Button».
2. Напрямна лінія. Описати алгоритм створення прямої руху. За допомогою яких елементів можна створити пряму?
3. Режим редагування шарів. Внесення змін до кадрів.
4. Порівняйте режими притягнення траєкторії до прямих та до середини об'єкту.

ЛІТЕРАТУРА

1. Дж. Лотт. Flash. Сборник рецептов. – СПб: Питер, 2007. – 544 с.
2. Б.Г. Жадаев. 100% самоучитель Macromedia Flash MX. – М.: Технолоджи-3000, 2005. – 544 с.
3. С. Jakson. Flash +. After Effects. – Amsterdam: Elsevier, 2008. – 277 p.
4. S. Webster, T. Yard, S. McSharry. Action Script 3.0 with Flash CS3 and Flex – California, USA: Friendcoft, 2008. – 550 p.

Лабораторна робота 3

Створення анкети з різними стилями елементів та відслідковуванням подій. Використання класу «Alert»

Мета роботи: закріпити навички створювати елементи з різними стилями, об'являти та визивати функції, відслідковувати події та створювати вікна, що впливають (з використанням класу «Alert»).

Елементи теорії. Елементи управління – це основні візуальні компоненти інтерфейсу користувача, такі як кнопки чи текст. Вони призначені для керування додатком при роботі користувача. Основні з них: «Button» – виглядає і функціонує, як справжня кнопка (можна використовувати в якості перемикача завдяки властивості «toggle»),

стан якої регулюється властивістю «selected»; «Label» – використовується, як мітка; «TextInput» – для введення строки тексту (властивість «text» дає можливість отримати, або задати значення строки, що відображається).

Функція – це фрагмент коду, що може бути використаний багаторазово. Розробник поміщує код ActionScript всередину функції, дає їй ім'я, а коли потрібно виконує цей код (посилається на функцію в необхідному місці).

Клас «Alert» відповідає за вивід вікна поверх додатку (за допомогою статичного методу «show()»).

ПОРЯДОК ВИКОНАННЯ РОБОТИ

1. У режимі «Design» створіть елементи «Button», «Label» та «TextInput» та довільно змініть їх стилі (на панелі «Style»). У режимі «Source» задайте для них атрибути «id» (тобто імена). Додайте тегу «Button» обробник подій «click» і помістіть в нього код на ActionScript, який буде описувати дії, що відбудуться при натисканні на кнопку;

Приклад:

```
<mx:Button id="Knop" click="Name.text  
='Ivanov' "/>
```

У результаті виконання даного коду, властивості «text» об'єкта «TextInput» (id="Name") буде присвоєне значення «Ivanov», після того, як користувач натисне кнопку.

2. Змініть властивості декількох об'єктів одночасно (наприклад, для «TextInput», «CheckBox») при натисканні на кнопку;

3. Виповніть ті ж самі дії, тільки з використанням функції, до якої звертається обробник подій «click»;

4. Використайте метод «setFocus ()» для одного з об'єктів на панелі;

5. Створіть функцію, що використовує клас «Alert» для виводу вікна поверх додатку. Імпорт класу виконується наступним чином:

```
import mx.controls.Alert;
```

Далі стоїть функція, в тілі якої необхідно використовувати метод «show()» класу «Alert», наприклад:

```
public function name():void
{
    Alert.show ("Текст" + Об'єкт.властивість);
}
```

6. Виведіть таким чином інформацію з декількох об'єктів: ім'я, прізвище, вік (задається користувачем через «NumericStepper»);

У результаті виконання роботи, ви отримаєте анкетну форму, при натисканні кнопок якої відбуваються різні дії: зміни властивостей об'єктів та поява вікон поверх додатку.

ЗМІСТ ЗВІТУ

1. Номер, тема та мета роботи.
2. Елементи теорії.
3. Програмний код розробленого додатку.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Середовище розробки додатків Flex. Основні напрямки використання технології.
2. Порівняйте середовища Flash і Flex (напрямки використання, мова програмування, інші допоміжні технології).
3. Панель «Style». Можливості зміни вигляду об'єкту у Flex.
4. Атрибути тегів. Для чого використовуються ідентифікатори об'єктів.
5. Створення власних функцій у середовищі Flex.
Синтаксис та основні конструкції.
6. Які події можуть бути відслідковані для елемента «Button»?
7. Опишіть алгоритм створення вікна за допомогою класу «Alert».

Література

1. А. Коул. Изучаем Flex3. Руководство по разработке насыщенных интернет-приложений. – С.П.: Символ, 2009. – 376 с.
2. С.Е. Brown. The Essential Guide to Flex with ActionScript 3.0. – California, USA: Friendssoft, 2007. – 483 p.
3. P. Elst, S. Jacobs, T. Yard. Object – Oriented Action Script 3.0. – California, USA: Friedsoft, 2007. – 599 p.
4. C. Moock. Essential Action Script 3.0. – Cambridge: O'Reilly, 2007. – 891 p.

Лабораторна робота 4

Відслідковування події «change». Використання зв'язаних даних

Мета роботи: удосконалити навички використання події «change». Навчитися виконувати явне перетворювання типу даних, багаторівневе зв'язування (модель даних) у тому числі з глобальними змінними. Використовувати тег `<mx:Binding/>`.

Елементи теорії. Подія «change» відбувається при виборі користувачем іншого об'єкту або зміні тексту елемента керування, наприклад «NumericStepper» чи «TextInput».

Метод зв'язування даних являється однією з переваг Flex – він дозволяє легко оперувати інформацією. Це простий метод звертання до даних, що дозволяє відслідковувати їх зміни. Він являє собою спосіб передачі і використання даних всередині додатку. Для збереження виразів зв'язування в окремому місці (за межами тегів компонентів MXML) використовують тег `<mx:Binding/>`. Однією з переваг цього тегу є те, що він дає можливість задавати декілька джерел прив'язування до одного адресату. Тег `<mx:Binding/>` може розташовуватись у будьякому місці коду між тегами відкриття та закриття

«Application», але тільки за межами контейнерів.

Flex також забезпечує зручність збереження структурованої інформації в моделі даних. Модель даних – це окремий об'єкт з деякою кількістю заданих властивостей, тобто спосіб збереження інформації в окремому місці. Його використання суттєво спрощує організацію коду. Вирази зв'язування, що використовуються всередині моделі, забезпечують багаторівневе зв'язування даних.

ПОРЯДОК ВИКОНАННЯ РОБОТИ

1. У режимі «Design» виберіть об'єкти «TextInput» та «Label». Задайте для них атрибути «id». Зв'яжіть данні з використанням фігурних дужок {} таким чином, щоб атрибут «text» об'єкту «Label» містив довільну строчку символів та інформацію з атрибуту «text» об'єкту «TextInput»;

2. Додайте до панелі об'єкт «NumericStepper» з обробником подій «change», що визиває функцію;

3. У тілі функції виконайте явне перетворення типу даних з «value» на «String» одним з трьох методів. Присвойте властивості «text» любого об'єкту (його можна зробити невидимим) значення, що було введене через «NumericStepper»;

Приклад:

```
TextInput_ім'я.text=  
NumericStepper_ім'я.value.toString()
```

4. Додайте отриману строчку до атрибуту «text» об'єкту «Label», що отриманий у 1-му пункті;

5. Створіть глобальну змінну типу «String» та зв'яжіть її з атрибутом «text» об'єкту «TextInput» за допомогою тегу <mx:Binding/>;

Приклад:

```
<mx:Binding destination="ім'я_змінної  
"source="input.text"/>
```

«Destination» – адресат прив'язування, «Source» – джерело прив'язування.

6. Виведіть на об'єкт «TextArea» данні, що були передані глобальній змінній;

7. Створіть модель даних (тег <mx:Model/>) з багаторівневим зв'язуванням;

В окремих тегах задайте ім'я, прізвище, по батькові, вік, адресу, телефон користувача та підготуйте для виводу деякі з цих даних, використовуючи зв'язування {}.

Приклад:

```
<mx:Model id="model">  
  <info>  
    <name>
```



```

    <firstname>
        {TextInput_им'я.text}
    </firstname>
    <secondname> Ivanov </secondname>
    <Person>{model.name.firstname}
        {model.name.secondname}
    </Person>
</name>
    <age>{NumericStepper_им'я.value}</age>
<RR> {model.name.Person}, your age
    {model.age}
</RR>
</info>
</mx:Model>

```

8. Використайте обробник подій «change» для зміни кольору додатку на той, що вибрав користувач за допомогою об'єкту «ColorPicker».

Приклад:

```

change="setStyle('backgroundColor',
ColorPicker_им'я.selectedColor)"

```

У результаті виконання роботи, ви отримаєте анкетну форму, в якій використовується модель даних, багаторівневе зв'язування даних та явне перетворення їх типу.

ЗМІСТ ЗВІТУ

1. Номер, тема та мета роботи.
2. Елементи теорії.
3. Програмний код розробленого додатку.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Типи даних у середовищі Flex. Глобальні змінні.

2. Скільки методів явного перетворення даних ви знаєте? Опишіть принаймні 2 методи.
3. Зв'язування даних за допомогою тегу `<mx:Binding/>`.
4. Модель даних. Створення моделі за допомогою тегу `<mx:Model/>`.
5. Покажіть на прикладі структуру моделі даних з багаторівневим зв'язуванням.
6. Подія «change». Виклик функцій за допомогою події «change».

ЛІТЕРАТУРА

1. А. Коул. Изучаем Flex3. Руководство по разработке насыщенных интернет-приложений. – С.П.: Символ, 2009. – 376 с.
2. М. Casario. Flex Solutions. Essential Techniques for Flex 2 and 3 Developers. – California, USA: Friendssoft, 2007. – 829 p.
3. С.Е. Brown. The Essential Guide to Flex with ActionScript 3.0. – California, USA: Friendssoft, 2007. – 483 p.
4. С. Kazoun, J. Loft. Programming Flex 2. – Cambridge: O'Reilly, 2007. – 463 p.

Лабораторна робота 5

Позиціонування дочірніх елементів

Мета роботи: засвоїти принципи роботи списку відображення, методів «addChild()» та «removeChild()», зміни розмірів об'єктів при натисканні на них.

Елементи теорії. Існує декілька способів розташування елементів додатку: можна встановлювати значення координат кожного елемента, використовувати контейнери розташування, обмежувачі або їх сполучення.

Списком відображення у Flex називається перелік усіх графічних елементів певного додатку, для якого діє принцип шарів: ті елементи, що розташовані у переліку вище (мають більш високий індекс) перекривають ті, що знаходяться нижче. Доступ до любого дочірнього об'єкта контейнера можна отримати по його індексу в переліку відображень.

Для додавання дочірніх елементів у контейнери використовується метод «addChild()», для видалення об'єкту зі списку – «removeChild()». Побажанню можливо легко поміняти розташування елементів у списку відображення. Спочатку для елемента використайте метод «removeChild()», а потім знову додайте його. В цьому випадку зміниться0 індекс елемента.

ПОРЯДОК ВИКОНАННЯ

1. У режимі «Design» додати на панель три контейнери «Canvas» різних кольорів. Позиціонування – абсолютне (див. рис. 1):

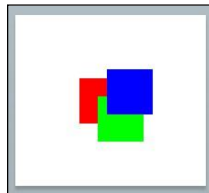


Рисунок 1 – Розміщення дочірніх елементів

2. Для кожного контейнера створити обробника подій, що викличе функцію, в тілі якої буде послідовно видалятися і знову додаватися той контейнер, на який натиснув користувач (для зміни його положення у списку відображення);

Для видалення елемента зі списку потрібно використовувати метод:
`colorsPanel.removeChild(Box_ім'я)`

Для додавання:

```
colorsPanel.addChild(Box_ім'я)
```

3. Необхідно створити змінну, яка вказує на те, що мета події – «Canvas»;

Приклад:

```
Var box:Canvas = event.currentTarget as Canvas
```

4. Задайте для кожного контейнеру атрибут «maxWidth». За умови, якщо в момент натискання ширина контейнеру була меншою за значення цього атрибуту, збільшить її на 10%, якщо навпаки – зменшить.

У результаті виконання роботи, ви отримаєте три контейнери «Canvas» різних кольорів. При натисканні на будь-який з них, він виходить на перший план і змінює свої розміри в залежності від виконання умови.

ЗМІСТ ЗВІТУ

1. Номер, тема та мета роботи.
2. Елементи теорії.
3. Програмний код розробленого додатку.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Позичіонування об'єктів у документі.
Способи позичіонування.

2. Абсолютне позиціонування. Особливості та встановлення властивостей об'єкту.
3. Контейнер «Canvas». Основні властивості та використання.
4. Методи «addChild()» та «removeChild()». Опишіть алгоритм використання методів на прикладі виконаної лабораторної роботи.
5. Зміна форми об'єктів. Опишіть на прикладі лабораторної роботи алгоритм зміни розмірів контейнерів.

ЛІТЕРАТУРА

1. А. Коул. Изучаем Flex3. Руководство по разработке насыщенных интернет-приложений. – С.П.: Символ, 2009. – 376 с.
2. К. Мук. ActionScript 2.0. Основы. – СПб.: Символ, 2006. – 576 с.
3. S. Jacobs. Flex for Developers. – California, USA: Friendsoft, 2008. – 534 p.
4. C. Kazoun, J. Loft. Programming Flex 2. – Cambridge: O'Reilly, 2007. – 463 p.
5. R. Blank, H. Otuome, O. Gonzalez. Advanced Flex Application Development Building. Rich Media. – California, USA: Friendsoft, 2008. – 475 p.

Лабораторна робота 6

Форми з перевіркою даних

Мета роботи: навчитися створювати форми з використанням валідаторів для перевірки введених даних, циклів та умовних операторів «if».

Елементи теорії. Flex дає можливість перевіряти введені дані за допомогою валідаторів. Основним з них являється `StringValidator` – інструмент перевірки, що підтверджує наявність введеного тексту. Для його використання необхідно розмістити тег `<mx:StringValidator/>` у верхній частині MXMLкоду у межах тегу `<mx:Application/>`. Валідатору присвоюється атрибут `source` для посилання на елемент, за яким необхідно вести контроль. Атрибут `property` вказує на властивість цього елемента. За допомогою властивості `requiredFieldError` можна змінити текст повідомлення про помилку. Якщо необхідно перевірити: чи має введений текст необхідну кількість символів, використовують властивість `to ShortError`.

Процес перевірки визивають тригери. Любий компонент валідації має властивість `trigger`, що посилається на елемент, за яким ведеться контроль. Для керування роботою декількох валідаторів у додатку, використовують допоміжну функцію класу `mx.validators.Validator - validateAll()`.

Умовний оператор `if` виконує частину коду при певних умовах: якщо передане йому значення – істинне.

Цикл дозволяє неодноразове виконання коду до тих пір, поки не виконається певна умова.

ПОРЯДОК ВИКОНАННЯ РОБОТИ

1. Створіть форму «Contact Editor Form». Додайте декілька елементів у контейнери «FormItem»: «TextInput», «HRule», «NumericStepper», «HBox» (містить «CheckBox», «Spacer», «Label»), поле «DataField» (встановити значення «true» властивості «editable» – для можливості введення дати з клавіатури та з календарю). Таким чином, будуть створені поля для ім'я, прізвища, віку, дати народження. Далі необхідне поле для введення телефону і його типу. Додайте

елемент «RadioButtonGroup» і створить три перемикача, що відповідають мобільному, домашньому і робочому телефонам.

Використовувати «CheckBox» в цьому випадку не коректно, оскільки користувачу можна вибрати лише один пункт. Для одного перемикача установити властивість «selected = true». Додайте поля для введення поштової адреси, mail та кнопку (її параметри будуть описані пізніше);

Форма може мати вигляд, приведений на рис. 1 а.

2. Для деяких контейнерів «FormItem» додайте властивість «required» зі значенням «true» (з'явиться зірочка, див. рис. 1 а). Це означає: для елементів буде проводитися перевірка на заповнення користувачем, але самі введенні данні перевірятися не будуть. Для цього використовуються валідатори;

3. Не виходячи за рамки тегу <mx:Application/> розмістіть тег <mx:StringValidator/>, присвойте йому атрибут «source», що вказує на елемент, над яким необхідно проводити контроль (в даному випадку це поля для ім'я та прізвища;

Приклад:

```
source="{id_елементу}"
```

4. За допомогою властивості «requiredFieldError» задайте текст повідомлення про помилку. Для переходу на наступний рядок у тексті повідомлення використовується комбінація символів «»;

5. Для поля, де задається прізвище, установіть властивість «required=false». У цьому випадку можна виконати перевірку на кількість введених символів. Установіть властивість «tooShortError»: помилка буде виводитися у випадку, якщо кількість символів менша ніж «minLength»;

6. Для перевірки даних з поля email використайте «EmailValidator». Текст про помилку введіть за допомогою властивості «requiredFieldError». Задайте властивість «trigger = "id_кнопки"» та встановіть «triggerEvent = "click"». Це означатиме, що перевірка почнеться після натискання кнопки;

Contact Editor

First Name *

Last Name

Age

Likes Dogs

Favorite Color

Data

tel.number mobile
 home
 work

mail *

Address

Button

Contact Editor

First Name * Введіть ім'я!

Last Name

Age

Likes Dogs

Favorite Color

Data

tel.number mobile
 home
 work

mail *

Address

Button

а

б

Рисунок 1 – Зовнішній вигляд форми

7. Для більшого контролю за валідацією використайте метод «validateAll()», що відноситься до класу «mx.validators.Validator» (базовий клас для всіх валідаторів). Цей метод має єдиний параметр, значенням якого є масив валідаторів;

Приклад:

```
private function validateAndSubmit():void
{
import mx.validators.Validator; var
validators:Array = [id,id,id,id]; var
errors:Array = Validator.
validateAll(validators);
}
```

8. Для відкриття вікна з інформацією для користувача про помилки використайте метод «show()» класу «Alert». Імпортуйте клас «mx.controls.Alert». Використайте умовний оператор «if» для визначення наявності помилок. Якщо вони є, то потрібно вивести повідомлення у вікні, що впливає;

Для цього використайте властивість «length», що визначає кількість елементів у масиві помилок (він повертається функцією «Validator.validateAll()»).

Приклад:

```
if(errors.length > 0)
{
Alert.show("текст повідомлення",
"строка заголовку вікна");
}
```

9. Для того, щоб виводилось повідомлення про всі помилки, необхідно використовувати цикл «for each...in», в якому обробити всі елементи масиву помилок, що являються результати перевірки «ValidationResultsEvents»;

10. Не раціонально для кожної помилки виводити окреме вікно. Їх можна зберегти в одному місці і потім відобразити в одному вікні. Для цього створіть масив «errorMessages» і додайте в нього повідомлення за допомогою методу «push()»:

```
For each (var error:ValidationResultEvent in
           errors)
{
var errorField:String = FormItem
(error.currentTarget.source.parent).label;
errorMessages.push(errorField+": "+
                    error.message); }
```

Таким чином, частина коду, що відповідає за контроль над елементами, повинна мати наступну структуру:

```
private function validateAndSubmit():void
{
    Імпорт класу валідаторів; Імпорт
    класу Alert;
    var validators:Array=[id,id,id,id]; var
    errors:Array = Validator.validateAll(
    validators);
    var errorMessages:Array=[];
    if(errors.length>0)
        { for each(var error:ValidationResultEvent
in errors)
            { var errorField:String=FormItem(error.
currentTarget.source.parent).label;
errorMessages.push(errorField+": "+error.mess
age);
            }
        Alert.show(errorMessages.join("\n\n"),
                    "There were problems");
    }
}
```

У результаті виконання роботи ви отримаєте форму, в якій при введенні даних відбувається перевірка. На рис. 1 б показана підказка для користувача. Також після того, як буде натиснута кнопка, відбудеться перевірка введених даних і у вікні з'явиться інформація про місця і характер помилок.

ЗМІСТ ЗВІТУ

1. Номер, тема та мета роботи.
2. Елементи теорії.
3. Програмний код розробленого додатку.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Які валідатори використовувалися у лабораторній роботі? Поясніть причину використання кожного з них.
2. Порівняйте можливості елемента «`RadioButtonGroup`» та «`CheckBox`».
3. Для чого використовується властивість «`required`». Яким чином перевіряються дані, якщо значення властивості «`true`»?
4. Використання класу «`mx.validators.Validator`». Метод «`validateAll()`».
5. Повідомлення про помилки. Вивід усіх повідомлень в одному вікні.
6. Для чого використовується властивість «`trigger`»?

ЛІТЕРАТУРА

1. К. Мук. ActionScript 2.0. Основы. – СПб.: Символ, 2006. – 576 с.
2. А. Коул. Изучаем Flex3. Руководство по разработке насыщенных интернет-приложений. – С.П.: Символ, 2009. – 376 с.
3. R. Blank, H. Otuome, O. Gonzalez. Advanced Flex Application Development Building. Rich Media. – California, USA: Friendsoft, 2008. – 475 p.
4. M. Casario. Flex Solutions. Essential Techniques for Flex 2 and 3 Developers. – California, USA: Friendsoft, 2007. – 829 p.

Лабораторна робота 7

Робота з текстом. Пошук збігу зі словником

Мета роботи: навчитися створювати компонент «TextInput», що буде підказувати користувачу варіант слова при заповненні.

Елементи теорії. Масив являє собою впорядковану сукупність об'єктів даних, що розглядаються як одне ціле. В ActionScript не існує масивів, як об'єктів окремого типу даних. Вони являються звичайними об'єктами, що відносяться до класу «Array». Елементами масиву ActionScript можуть бути довільні об'єкти: числа, строки, об'єкти чи масиви. Створити масиви можна за допомогою функції-конструктора та ключового слова new. Особливістю функції «Array» є те, що вона повертає масив, навіть якщо оператор «new» не був задіяний.

Змінні використовуються для збереження інформації і наступного використання у додатку. В ActionScript змінні задаються за допомогою виразу «var», за яким стоїть ім'я змінної та присвоєне значення. Змінні мають різний рівень доступу, як функції.

ПОРЯДОК ВИКОНАННЯ РОБОТИ

1. Задайте елементу «TextInput» подію «change», що визиває функцію перевірки введення та пошуку у словнику;
2. У тілі функції створіть масив, елементами якого являються слова серед яких буде виконуватися пошук;
3. Також створіть змінну, якій буде присвоюватись введений текст, як шаблон:

```
private var sravnenie:RegExp;
```

4. У тілі функції, що визивається обробником подій «change», створіть масив з елементів, які пройшли порівняння:

```
var temp: Array=all.filter(filter);
```

та присвойте властивості «text» об'єкта «TextInput» елемент цього масиву.

5. Використайте функцію «filter», яка дозволяє створити метод, що отримує довільний об'єкт і після деякого опрацювання повертає логічний признак включення у відфільтрований масив:

```
private function filter(element:*,
    index:int, arr:Array):Boolean
```

6. У тілі даної функції змінній «sравnenie» присвойте нове значення шаблону:

```
sравnenie=new RegExp(input.text);
```

та поверніть результат порівняння у вигляді строчки:

```
return(sравnenie.test(element as String));
```

У результаті виконання роботи, ви отримаєте об'єкт «TextInput». При введенні тексту у поле, будуть з'являтися підказки (схожі слова зі словника).

ЗМІСТ ЗВІТУ

1. Номер, тема та мета роботи.
2. Елементи теорії.
3. Програмний код розробленого додатку.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Компонент «TextInput». Основні властивості та методи.
2. Тип змінних «RegExp». Використання шаблонів у додатках.
3. Фільтрування даних в масиві за допомогою функції «filter()».
4. Опишіть алгоритм фільтрування даних зі словника.

ЛИТЕРАТУРА

1. Дж. Ноубл, Т. Андерсон. Flex3. Сборник рецептов. – Спб.: Символ, 2009. – 736 с.
2. А. Коул. Изучаем Flex3. Руководство по разработке насыщенных интернет-приложений. – С.П.: Символ, 2009. – 376 с.
3. R. Blank, H. Otuome, O. Gonzalez. Advanced Flex Application Development Building. Rich Media. – California, USA: Friendsoft, 2008. – 475 p.
4. Flex 2. Programming Action Script 3.0./ – California, USA: Adobe Systems Incorporated, 2006. – 515 p.

Лабораторна робота 8

Складні списки даних. Використання даних у форматі XML. Завантаження зовнішніх даних у процесі компіляції

Мета роботи: навчитися використовувати масиви об'єктів, в якості складних списків даних; елементи «DataGrid»; дані у форматі XML; створювати XML-файли та підключати їх до додатку.

Елементи теорії. У якості простих списків можуть виступати масиви строк. Якщо до елемента управління списками потрібно передати об'єкти з різними властивостями, доцільно використовувати складні списки – масиви об'єктів. Тоді для елементів списку необхідно використовувати властивість «labelField», що визначає яку саме властивість об'єкту зі складною структурою потрібно відобразити. Якщо необхідно відображати декілька параметрів одночасно, використовують «DataGrid». Він дозволяє створювати декілька колонок, кожна з яких відповідає властивості об'єкту. Замість масивів простих елементів і об'єктів можна використовувати дані у форматі XML (для цього використовується тег <mx:XML/>).

ПОРЯДОК ВИКОНАННЯ РОБОТИ

1. Створіть масив (тег <mx:Array/>) та передайте його до елемента управління списком «List» за допомогою властивості «dataProvider»;

Приклад:

```
<mx:dataProvider>
  <mx:Array>
    <mx:Object> song="name"
                album="name"
                artist="name"
    </mx:Object>
  </mx:Array>
</mx:dataProvider>
```

У цьому випадку відобразатись будуть тільки параметри, що визначені за допомогою властивості «labelField» елемента «List» (наприклад, виведеться список «song»).

2. Для того, щоб відобразити відразу декілька властивостей об'єкту використайте елемент «DataGrid» (схожий на електроні чи HTML - таблиці). Створіть його в режимі «Design». Кількість колонок визначається властивістю «columns», якій присвоюється послідовність елементів «DataGridColumn». У кожному з них необхідно вказати властивість для відображення за допомогою атрибута «dataField»;

Приклад:

```
<mx:DataGrid textAlign="center"
    borderThickness="20" >
    <mx:columns>
    <mx:DataGridColumn headerText="name"
        dataField="song"/>
    <mx:DataGridColumn headerText="name"
        dataField="album"/>
    <mx:DataGridColumn headerText="name"
        dataField="artist"/>
    </mx:columns> </mx:DataGrid
```

У результаті ви отримаєте таблицю, в якій можливо переставляти стовбці чи строчки, сортувати. При великому обсязі інформації з'являються полоси прокрутки (див. рис. 1).

ID	First	Last
1	Masha	Ivanova
0	Petya	Sidorov

Рисунок 1 – Вигляд елемента «DataGrid»

3. Запишіть дані у форматі XML. Для цього використовуйте тег <mx:XML/>. Створіть кореневий тег <contacts/>, який обов'язково

необхідний для XML-документу. Він може включати в себе довільну кількість вкладених тегів <contact/>, кожен з яких має атрибут «id», що слугує в якості числового ідентифікатора контакту. Кожний тег <contact/> може мати довільну кількість дочірніх елементів, що відповідають за окремі властивості;

Приклад:

```
<contacts>
  <contact id="0">
    <firstName>Petya</firstName>
    <secondName>Sidorov</secondName>
    ...
  </contact>
  <contact id="1">
    <firstName>Masha</firstName>
    <secondName>Ivanova</secondName>
    ...
  </contact>
</contacts>
```

Як інформація можуть використовуватися поля з форми «Contact Editor» (див. лабораторну роботу 6).

4. Прив'яжіть властивість «dataProvider» елементу «DataGrid» до XML-документу, використовуючи фігурні дужки:

```
<mx:DataGrid id="contacts"
  dataProvider="{XML_name.contact}">
```

Важливо, що для першої колонки у таблиці «DataGrid» значенням властивості «dataField» є не просто «id», а «@id», оскільки в даному випадку це атрибут XML, а не дочірній елемент.

5. Створіть XML файл. Для цього виберіть «File → New → File». Збережіть його у папку з вихідним кодом поточного проекту (папку src) і назвіть «contacts.xml». Скопіюйте дані, що розміщені всередині тега <mx:XML/> в цей файл;

6. Для тегу `<mx:XML/>` виставте властивість `source = "contacts.xml"`.

Тепер під час компіляції з файлу «contacts.xml» будуть зчитуватися дані і виводитися у таблиці «DataGrid».

ЗМІСТ ЗВІТУ

1. Номер, тема та мета роботи.
2. Елементи теорії.
3. Програмний код розробленого додатку.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Опишіть структуру даних у форматі XML на прикладі. За допомогою якого тегу можливо записати дані у форматі XML?
2. Які елементи управління списками даних ви знаєте?
3. Як створити XML-файл за допомогою Flex?
Використання XML-файлів у додатку.
4. Для чого використовується властивість «dataProvider» елемента «DataGrid»?
5. Для чого використовується властивість «labelField» елемента «List»?
6. Що таке точкова нотація в ActionScript?

ЛІТЕРАТУРА

1. А. Коул. Изучаем Flex3. Руководство по разработке насыщенных интернет-приложений. – С.П.: Символ, 2009. – 376 с.
2. S. Webster, T. Yard, S. McSharry. Action Script 3.0 with Flash CS3 and Flex – California, USA: Friendcoft, 2008. – 550 p.
3. С.Е. Brown. The Essential Guide to Flex with ActionScript 3.0. – California, USA: Friendsoft, 2007. – 483 p.

Лабораторна робота 9

Завантаження зовнішніх даних при запуску додатку

Мета роботи: навчитись використовувати тег `<mx:HTTPService/>` для доступу до текстових та XML-документів через мережу Інтернет по стандартному протоколу HTTP (Hypertext Transfer Protocol).

Елементи теорії. Для великих додатків доцільніше використовувати XML-файл (розміщується в папці «src») ніж XML-код у тілі програми. Доступ до таких файлів через мережу Інтернет по протоколу HTTP можна отримати за допомогою тегу `<mx:HTTPService/>`. `HTTPService` має властивість «url», яка дозволяє вказати шлях до потрібного файлу.

Запуск компоненти сервесу виконується через метод «send()».

ПОРЯДОК ВИКОНАННЯ РОБОТИ

1. Створіть елемент «DataGrid» аналогічний тому, що використовувався у лабораторній роботі 8;

2. Використовуючи тег `<mx:HTTPService/>` здійсніть доступ до текстового файлу (аналогічного тому, що використовувався у лабораторній роботі №8) через протокол HTTP. Для цього необхідно використовувати відносну URL-адресу, оскільки файл являється частиною вашого проекту. У даному випадку `url="contacts.xml"`;

3. Помістіть наступний код у рамках тегу `<mx:Application/>`:

```
<mx:HTTPService id="contactsService"
resultFormat="e4x" url="contacts.xml"/>
```

Таким чином, буде створюватись компонент `HTTPService`, що посилається на файл «contacts.xml». Необхідно уточнювати властивість «resultFormat», оскільки по замовчуванню приймається значення «object» і отримані дані будуть перетворюватись компонентом в об'єкти. Тому, укажіть значення «resultFormat - e4x». Це буде означати: завантажені данні знаходяться у форматі XML. «e4x» розшифровується, як ECMAScript for XML – технологія, що схожа на

точкову нотацію в ActionScript (використовувалась у лабораторній роботі 8).

4. Для властивості «dataProvider» елементу «DataGrid» необхідно встановити інше значення ніж те, що використовувалось у лабораторній роботі 8. Для вказування посилання на дані, що отримані з сервера, використовується властивість сервісу «lastResult». Її значення вміщує дані, що отримані у результаті останнього виклику даного сервісу:

```
dataProvider="{contactsService.lastResult.  
contact}"
```

Властивість «contactsService.lastResult» містить дані, що повертаються сервером, у форматі XML. Однак, у «dataProvider» необхідно вказати точне посилання на список контактів. Для цього використовується вираз: contactsService.lastResult.contact.

5. Для виклику компонента сервісу необхідно використовувати метод «send()». У межах тега <mx:Application/> встановіть обробник подій «applicationComplete», що викличе необхідний метод;

Приклад:

```
applicationComplete="contactsService.send()"
```

Тепер, при запуску додатку, ініціюється робота сервісу, і отримані дані будуть відображені в таблиці «DataGrid» за допомогою технології зв'язування. Цей метод використовується і для отримання даних через мережу Інтернет. Наприклад, можна використати інформацію з XML-файлу, що знаходиться на сервері за адресою (url): <http://greenlike.com/flex/learning/projects/contactmanager/contacts.xml>.

ЗМІСТ ЗВІТУ

1. Номер, тема та мета роботи.
2. Елементи теорії.
3. Програмний код розробленого додатку.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Протокол ННТР. Опишіть алгоритм роботи протоколу.
2. Використання тегу `<mx:HTTPService/>`. Атрибут «url».
3. Використання тегу `<mx:HTTPService/>`. Атрибут «resultFormat».
4. Використання подія «applicationComplete».

ЛІТЕРАТУРА

1. К. Мук. ActionScript 2.0. Основы. – СПб.: Символ, 2006. – 576 с.
2. С. Kazoun, J. Loft. Programming Flex 2. – Cambridge: O'Reilly, 2007. – 463 p.
3. S. Jacobs. Flex for Developers. – California, USA: Friendssoft, 2008. – 534 p.

Лабораторна робота 10

Підключення результатів пошуку до додатку

Мета роботи: навчитись створювати прості компоненти для доступу до пошукового сервісу Yahoo, що дозволить розміщувати відпрацьовані дані у Flex-додатку.

Елементи теорії. Для доступу до пошукового сервісу використовується бібліотека Yahoo! ASTRA WebAPIs, яку необхідно додати до папки «libs». Після чого, відразу стане доступним компонент «SearchService». Для його правильної роботи необхідно задати лише одну властивість – текстову строку, що визначає критерій пошуку за допомогою атрибуту «query». «SearchService» використовує ті самі методи і властивості, що і HTTPService.

Елементи списку Flex мають вбудовану підтримку переміщення розташованих в них даних, яка реалізована за допомогою властивостей «dragEnabled» та «dropEnabled».

ПОРЯДОК ВИКОНАННЯ РОБОТИ

1. Для створення додатку вам знадобиться ActionScript 3 Search API. Використовуйте бібліотеку «AstraWebAPIs.swc». Додайте цей файл у папку «libs» вашого додатку;
2. Для правильної роботи «SearchService» додайте атрибут «query», що визначає критерій пошуку. Значення цієї властивості можна прив'язати до елемента «TextInput»;
3. Створіть об'єкт «Button». При натисканні визвіть метод «send()» компоненту «SearchService» (він також має властивість «lastResult»);
4. При створенні додатку, використовуйте наступний код:

```
<mx:Application layout="absolute"
  xmlns:mx="http://www.adobe.com/2006/mxml"
  xmlns:yahoo="http://www.yahoo.com/astra/
    2006/mxml">
<yahoo:SearchService id="searchService" applicat
ionId="YahooDemo"query="{TextInput_id.text}"/>
</mx:Application>
```

5. Всередині тегу `<mx:Application/>` створіть контейнер «HBox», в ньому дочірні елементи: «Button» (див. попередній пункт); «TextInput» у рамках тегу `<mx:FormItem/>`;

Приклад:

```
<mx:FormItem label="Query:">
    <mx:TextInput id="queryTextInput"/>
</mx:FormItem>
```

Для контейнера HBox використовуйте властивість «defaultButton», що вказує на кнопку для якої відбудеться подія «click», при натисканні користувачем клавіші «Enter» (оскільки, часто замість кнопки натискається «Enter»).

Приклад:

```
defaultButton="{Button_id}"
```

6. У межах тегу `<mx:Application/>` створіть об'єкт «List» (аналогічний тому, що використовувався у лабораторній роботі №9). Додайте властивість «showDataTips="true"», яка керує появою підказок при наведенні користувачем на посилання;

7. Елементи списку Flex підтримують можливість переміщення даних, що в них розташовані. Це реалізується за допомогою властивостей «dragEnabled» та «dropEnabled». Створеному об'єкту «List» присвойте властивість «dragEnabled="true"». Це дозволить переносити елементи з цього списку в інший;

8. Створіть контейнер «Panel» з вкладеним елементом «List», для якого установіть властивість «dropEnabled="true"». Це дозволить переміщувати об'єкти в цей список;

Запустіть додаток. Введіть дані у строку пошуку. Спробуйте перетягнути посилання до іншого списку.

9. Для налаштування вигляду елементів списку використовується мітка «itemRenderer». Застосуйте цей об'єкт для першого списку (id="resultList"). Замість властивостей: «showDataTips», «labelField», «dragEnabled» використовуйте наступну частину коду:

```

    <mx:itemRenderer>
      <mx:Component>
        <mx:VBox width="100%">
<mx:Label text="{data.name}" fontWeight="bold"/>
<mx:Text width="100%" text="{data.summary}"/>
<mx:Text width="100%" text="{data.clickURL}"/>
        </mx:VBox>
      </mx:Component>
    </mx:itemRenderer>

```

Тег `<mx:Component/>`, в середині якого знаходиться MXML-міст вашого `<mx:itemRenderer/>`, вказує на те, що Flex має створити компонент і у подальшому застосовувати його до кожного пункту списку. Замість властивості `«labelField»` використовується елемент `«Label»`. Прив'язування до властивості `«data»` вказує на елемент для відображення.

Результати пошуку являються об'єктами класу `«WebSearchResult»`, що включений до бібліотеки `ASTRA Web APIs`. Він має такі властивості: `«name»`, `«summary»` (короткий огляд змісту сторінки), `«clickURL»` (URL-адреса сторінки).

Тому, можливо виконати прив'язування до необхідних властивостей. Оскільки, для відображення отриманих даних знадобиться більше однієї строчки, доцільно використовувати елемент `«Text»`. Для сумісного використання елементів `«Label»` та `«Text»`, їх можна помістити до контейнеру `«Vbox»`.

10. Для того, щоб список результатів пошуку виглядав краще, можна використати властивість `«variableRowHeight="true"»`. Це дозволить варіювати висоту рядів, щоб умістити необхідну інформацію за певним посиланням.

У результаті виконання роботи отримаєте додаток, що містить список знайдених у мережі Інтернет сторінок, короткий огляд їх змісту, URL-адреси.

ЗМІСТ ЗВІТУ

1. Номер, тема та мета роботи.
2. Елементи теорії.

3. Програмний код розробленого додатку.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Використання сторонніх бібліотек у Flex додатках. Способи підключення бібліотек.
2. Використання властивості «defaultButton» для контейнера «HBox».
3. Використання властивості «showDataTips» для об'єкта «List».
4. Використання властивості «variableRowHeight» для елемента «Text».
5. Переміщення даних між списками. Властивості «dragEnabled» та «dropEnabled».

ЛІТЕРАТУРА

- 1 R. Blank, H. Otuome, O. Gonzalez. Advanced Flex Application Development Building. Rich Media. – California, USA: Friendsoft, 2008. – 475 p.
2. C. Kazoun, J. Loft. Programming Flex 2. – Cambridge: O'Reilly, 2007. – 463 p.
3. S. Jacobs. Flex for Developers. – California, USA: Friendsoft, 2008. – 534 p.

Лабораторна робота 11

Компонент навігації «TabNavigator». Створення галереї

Мета роботи: закріпити знання про використання HTTPService, навчитись створювати додатки з галереєю за допомогою компонента навігації «TabNavigator» та використовувати елемент «ProgressBar».

Елементи теорії. Іноді вільного простору для розміщення усіх необхідних елементів додатку може виявитись не достатньо. У такому випадку потрібно вирішити: які компоненти будуть видимими одночасно, або створити можливість вибору між декількома видами всередині додатку. До стандартного набору Flex входять компоненти, що дозволяють розробнику легко керувати поточним відображенням додатку і створювати різні види. Вони називаються контейнерами навігації і призначені для організації переключення між дочірніми елементами, причому видимим буде лише один. Зовнішнє переключення між окремими видами відбувається за допомогою вкладок. Найчастіше використовується контейнер «TabNavigator». Він може включати в себе скільки завгодно вкладених елементів. У якості дочірніх – можуть виступати лише контейнери. Кожен з них має властивість «label», яка відображається на відповідній вкладці.

Для того, щоб механізм роботи вашого додатку був зрозумілим користувачу, можна використовувати елемент «ProgressBar» для відображення ходу процесу завантаження зображення до галереї. Достатньо лише присвоїти властивості «source» ім'я (id) зображення.

Контейнер «TileList» може відображати зменшенні зображення для попереднього перегляду. Для реалізації цієї можливості знадобиться «itemRenderer» та ще один елемент «Image».

ПОРЯДОК ВИКОНАННЯ РОБОТИ

1. У режимі «Design» перенесіть на сцену елемент «TabNavigator» – контейнер для реалізації зміни видів. Він може включати в себе необмежену кількість елементів, кожен з яких отримує відповідну вкладку. У якості дочірніх елементів можуть виступати лише контейнери, які мають властивість «label»;

Після створення «TabNavigator» буде містити єдиний дочірній елемент «Canvas».

2. Виділіть «TabNavigator» у режимі «Design». З'явиться додаткова панель, що містить кнопки для додавання та видалення дочірніх контейнерів;

3. Натисніть кнопку «+». У діалоговому вікні, що з'явилося, виберіть тип контейнеру «Canvas». Для отриманих двох дочірніх елементів виставте мітки «List View» та «Tile View» (плитка);

4. У дочірні елементи додайте елементи «List» та «TileList». Виставте їх параметри таким чином, щоб вони займали весь вільний простір;

Оскільки фотоальбом має список фотографій, то можна використовувати структуровані за допомогою XML дані.

5. Створіть файл «photos.xml» і помістіть у папку «src». Доступ до нього виконайте за допомогою HTTPService;

6. У файл помістіть наступний код:

```
<photos> title="Crowdy Head Lighthouse"
  thumb="http://www.greenlike.com/photo
  gallery/lighthouse_thumb.jpg"
  image="http://www.greenlike.com/photo
  gallery/lighthouse.jpg" />
  <photo title="Uluru"
thumb="http://www.greenlike.com/
  photogallery/uluru_thumb.jpg"
  image="http://www.greenlike.com/photo
  gallery/uluru.jpg"/>
  <photo title="Karnak Temple"
thumb="http://www.greenlike.
  com/photogallery/temple_thumb.jpg"
  image="http://www.greenlike.com/photo
  gallery/temple.jpg"/>
  <photo
  title="Contemplating the Purchase"
  thumb="http://www.greenlike.com/photo
  gallery/purchase_thumb.jpg"
  image="http://www.greenlike.com/photo
  gallery/purchase.jpg"/>
</photos>
```

Даний код являє собою перелік фотографій, кожна з яких має атрибути: «title», «thumb» (посилання на зменшену копію зображення для попереднього перегляду), «image» (URL-адреса повноформатного зображення).

Для того, щоб використовувати власні зображення, помістіть їх у папку з вихідним кодом і змініть URL-адресу в XML-файлі.

7. Установіть обробник подій «applicationComplete», що викликає метод «send()» для ініціалізації сервісу по закінченню завантаження;

8. Для елемента «List» властивість «dataProvider» прив'яжіть до «service.lastResult.photo». Властивості «labelField» присвойте значення «@title» (оскільки, назва фотографії представлена XML-атрибутом, то доступ до нього можливий через E4Xвираз);

9. Створіть елемент «Image». Атрибут «source» прив'яжіть до «photosList.selectedItem.@image». Вирівнювання виставте по центру;

10. Додайте у режимі «Design» елемент «ProgressBar», який буде відображати процес завантаження зображення у відсотках;

11. Для того, щоб елемент «ProgressBar» був видимий лише під час завантаження і зникав по закінченню цього процесу, необхідно керувати його властивістю «visible». Для події «open» елемента «Image» установіть значення «progressbar.visible=true»; для події «complete» – «progressbar.visible=false». Для «ProgressBar» установіть властивість «visible="false"»;

Тобто, по закінченню завантаження зображення, «ProgressBar» буде зникати.

Потрібно налаштувати також елемент «TileList», який має можливість попереднього перегляду зменшених зображень. Для цього знадобиться «itemRenderer» в елементі «TileList» і ще один елемент «Image».

12. Властивості «source» об'єкту «Image», що використовується для перегляду зменшених копій, присвойте значення «data.@thumb»;

13. Можливо також забезпечити появу підказки з назвою зображення при наведенні курсору. Для цього використовуйте властивість «toolTip», яку прив'яжіть до «data.@title»;

Необхідно забезпечити синхронізацію роботи списків. Це можна

реалізувати за допомогою спеціальної властивості «creationPolicy», що приймає одне з чотирьох значень «all», «auto», «queued» та «none». По замовчуванню використовується «auto» – в цьому випадку завантажуватись буде лише один вигляд. Значення «all» створює усі види; «queued» – створює дочірні контейнери, а потім всі вкладені елементи по черзі; «none» – відміняє створення будь-яких видів.

14. Установіть для властивості «creationPolicy» значення «all».

У результаті виконання роботи, ви отримаєте додаток з галереєю, що має дві вкладки і дозволяє переглядати зображення у режимі списку чи плитки. Для більшої наочності можете використати різні візуальні та звукові ефекти, фільтри (наприклад, «DropShadowFilter») для об'єкту «Image».

ЗМІСТ ЗВІТУ

1. Номер, назва та мета роботи
2. Елементи теорії
3. Код створеного додатку

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Елемент «TabNavigator». Його використання та основні властивості.
2. Синхронізація роботи списків. Методи синхронізації.
3. Елемент «ProgressBar». Використання елемента в інтерфейсі користувача.
4. Яким чином забезпечується перегляд зменшених зображень у даній роботі?

ЛІТЕРАТУРА

1. А. Коул. Изучаем Flex3. Руководство по разработке насыщенных интернет-приложений. – С.П.: Символ, 2009. – 376 с.
2. S. Webster, T. Yard, S. McSharry. Action Script 3.0 with Flash CS3 and Flex – California, USA: Friendcoft, 2008. – 550 p.
3. S. Jacobs. Flex for Developers. – California, USA: Friendsoft, 2008. – 534 p.
4. C. Kazoun, J. Loft. Programming Flex 2. – Cambridge: O'Reilly, 2007. – 463 p.

Лабораторна робота 12

Використання таблиць стилів. Синтаксис CSS

Мета роботи: ознайомитись з поняттям таблиць стилів, синтаксисом CSS. Навчитись використовувати таблиці стилів із зовнішніх файлів.

Елементи теорії. Зовнішній стандартний вигляд компонентів, що використовуються при створенні додатків, може бути кардинально змінений за допомогою різних стилів. Розробник за допомогою каскадних таблиць стилю (CSS) може створити стиль, що буде використаний для багатьох об'єктів додатку, або навіть для багатьох додатків. CSS мають механізм наслідування: стиль, що встановлений для контейнеру (або «Application») передається його внутрішнім елементам. CSS – це окрема мова, що не пов'язана з XHTML та ActionScript. Вона дозволяє описувати стилі компонентів не у відповідних тегах, а в окремих файлах, які зв'язані з додатком за допомогою

`<mx:Style/>`. В основі синтаксису CSS лежить правило стилю (його ім'я), за яким у фігурних дужках стоїть опис стилю (властивість стилю: її значення). Опис CSS стилів можна створювати трьома способами: за допомогою селекторів класу, селекторів типу, глобальних стилів.

Селектори класу – це правила стилів з певним ім'ям, що можуть застосовуватись по відношенню до будь-якого компоненту. Приклад CSS-коду:

```
<mx:Style> text
{
  color: #FFFFFF; fontSize:
  18;
}
</mx:Style>
```

Селектори типу дозволяють визначити стиль для усіх компонентів певного типу. Приклад CSS-коду:

```

<mx:Style> Button
{
  color: #FFFFFF; fontSize:
  18;
}
</mx:Style>

```

За допомогою глобальних стилів можна застосувати певні стилі до всього додатку в цілому. Відміна CSS-коду від попереднього полягає лише в тому, що не потрібно вказувати тип компонентів. Замість цього пишеться слово «global».

ПОРЯДОК ВИКОНАННЯ РОБОТИ

1. Створіть додаток схожий на «ContactManager» (див. лабораторну роботу №6). Використовуйте такі об'єкти, як «DataGrid», «dataGridHeader», «Panel», «panelTitle», «TextInput»;
2. Створіть CSS-файл: «File → New → CSS File»;
3. У додатку додайте тег, що буде імпортувати цю таблицю стилів: `<mx:Style source="ім'я_файлу .css"/>`;
4. Запішіть у файл CSS-код, що буде описувати стилі різних об'єктів;

Наприклад, для глобального стилю виставте властивість «backgroundAlpha» (прозорість) і деяке її значення, а також певний шрифт. Для додатку в цілому змініть властивості «backgroundColor» та «themeColor». Для об'єкту «DataGrid» налаштуйте такі властивості стилів: «alternatingItemColors» (кольори полів, що перемжні через один), «backgroundDisabledColor», «headerColors», «headerStyleName» та ін. Для «Panel» налаштуйте параметри: «backgroundAlpha», «backgroundColor», «borderAlpha» (прозорість межі), «borderColor», «cornerRadius». Довільно налаштуйте параметри «panelTitle», «TextInput».

У результаті виконання роботи, ви отримаєте додаток з вашими налаштуваннями властивостей стилів, які можна змінювати при завантаженні різних CSS-файлів.

ЗМІСТ ЗВІТУ

1. Номер, назва та мета роботи
2. Елементи теорії
3. Код створеного додатку

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Що таке каскадні таблиці стилів?
2. Як пов'язати додаток з CSS-файлом?
3. Назвіть та опишіть способи опису CSS-стилів.
4. Для чого використовується властивість `cornerRadius`?

ЛІТЕРАТУРА

1. R. Blank, H. Otuome, O. Gonzalez. Advanced Flex Application Development Building. Rich Media. – California, USA: Friendsoft, 2008. – 475 p.
2. С.Е. Brown. The Essential Guide to Flex with
ActionScript 3.0. – California, USA: Friendsoft, 2007. – 483 p.
3. M. Casario. Flex Solutions. Essential Techniques for Flex 2 and 3 Developers. – California, USA: Friendsoft, 2007. – 829 p.

Лабораторна робота 13

Створення меню додатків

Мета роботи: ознайомитись з принципами роботи класу MenuBar для створення багаторівневих каскадних меню додатків.

Елементи теорії. Для створення додатків зі складною логічною будовою та зрозумілим для користувача інтерфейсом необхідно реалізувати механізми навігації та управління. Для цього можуть використовуватися багаторівневі каскадні меню з розгалуженою системою наслідування. У середовищі Flex меню виводиться на екран за допомогою тегу `<mx:MenuBar/>`. Одним із його атрибутів є «dataProvider», який зберігає структуру меню. Для того, щоб побудувати меню використовуються клас «XMLListCollection», який у форматі XML представляє ієрархію пунктів за допомогою тегів `<menuitem/>`.

Приклад:

```
<menuitem label="SubMenuItem" type="radio"
groupName="one" data="текст"/>
```

Атрибут «label» задає назву пункту меню. Атрибут «type» дозволяє задавати елементи управління до пункту меню: наприклад, «radio» – має вигляд, як «RadioButton». Атрибут «groupName», в даному випадку, прив'язує пункт меню до групи «one».

ПОРЯДОК ВИКОНАННЯ РОБОТИ

1. У блоці CDATA підключіть наступні бібліотеки:

```
mx.events.MenuEvent;
mx.controls.Alert;
mx.collections.*;
```

2. За допомогою «XMLList» створіть ієрархічну структуру вашого меню. Для цього використовуйте приватну змінну «menubarXML» типу «XMLList»;

Приклад:

```
private var menubarXML:XMLList =
    <>
        Структура вашого меню
    </>;
```

3. Створіть три-рівневе меню вашого додатку за схемою;

```
<menuitem label="Menu1">
    <menuitem label="MenuItem" data="текст"/>
    <menuitem label="MenuItem" data="текст"/>
</menuitem>
```

4. За допомогою приватної функції «initCollections()», що викликається по закінченню побудови додатку, створіть змінну для атрибуту «dataProvider» тегу <mx:MenuBar/>;

Приклад:

```
private function initCollections():void {
    menuBarCollection = new
        XMLListCollection(menubarXML);
}
```

5. Додайте обробник подій, що буде реагувати на вибір пункту меню користувачем та виводити за допомогою класу «Alert» значення атрибуту «data».

Приклад:

```
function menuHandler(event:MenuEvent):void {
    Alert.show("Назва:"+event.item.@label+"\n"+
        "Данні:"+event.item.@data,"Заголовок
        повідомлення");
}
```

ЗМІСТ ЗВІТУ

1. Номер, назва та мета роботи
2. Елементи теорії

3. Код створеного додатку

КОНТРОЛЬНІ ЗАПИТАННЯ

1. За допомогою якого тегу у середовищі Flex будується меню?
2. Які основні атрибути має тег для створення меню?
3. Як створити змінну типу «XMLList»?
4. Який тип даних передається в атрибут «dataProvider» тегу для створення меню?

ЛІТЕРАТУРА

1. R. Blank, H. Otuome, O. Gonzalez. Advanced Flex Application Development Building. Rich Media. – California, USA: Friendsoft, 2008. – 475 p.
2. S. Jacobs. Flex for Developers. – California, USA: Friendsoft, 2008. – 534 p.
3. R. Blank, H. Otuome, O. Gonzalez. Advanced Flex Application Development Building. Rich Media. – California, USA: Friendsoft, 2008. – 475 p.

Навчальне видання

Методичні вказівки до виконання лабораторних робіт

з курсу «Візуальні мови програмування» для студентів напряму
підготовки 171 «Електроніка» денної та заочної форми навчання

Відповідальний за випуск І.Ю. Проценко

Редактор _____

Комп'ютерне верстання С.І. Проценко

Підп. до друку _____, поз.30

Формат 60x84/16. Папір офс. Гарнітура Times New Roman Суг. Друк
офс.

Ум. друк. арк. Обл.-вид. арк.

Тираж 50 пр. Собівартість вид.

Зам. №

Видавець і виготовлювач

Сумський державний університет

4007, м. Суми, вул. Р.-Корсакова, 2 Свідоцтво суб'єкта видавничої справи ДК
№3062 від 17.12.2007.