بسم الله الرحمن الرحيم

GUI in j2m

أسماء الطلبة:-

كرار فالح حسن

علي خالد حسين

مصطفى جواد كاظم

مصطفى طلال محمد

علي قائد كريم

Introduction

- -J2ME GUI allows you to develop mobile applications with the look-and-feel, as well as functionality of desktop applications. Mobile Java developers are well aware of the severe limitations of the built-in user interface provided by J2ME. The interface is completely at the mercy of the virtual machine with little or no control left to the developer. This is fine for disposable small applications, but most commercial and larger applications require exact control over their user interfaces, even if only for aesthetic purposes.

Introduction

- Developers only have two real options: Develop their own user interface using low-level graphical instructions, or use a third-party mobile GUI. There are only a few mobile GUI libraries worth considering, but most of them fall victim to some of the following problems:
- Many mobile GUI libraries offer components which are as visually unappealing as that of the built-in user interface of J2ME.
- Several compatibility issues with many of the GUI libraries. Some claim to be compatible with almost all mobile handsets whereas in reality they only work on a small subset of handsets.
- The few GUI libraries with acceptable interfaces are usually very bulky and are actually more geared towards newer handsets. Their large JAR file sizes leave little room, if any, for the actual application and some suffer from performance issues on older handsets.

Introduction

- The few GUI libraries with acceptable interfaces are usually very bulky and are actually more geared towards newer handsets. Their large JAR file sizes leave little room, if any, for the actual application and some suffer from performance issues on older handsets.
- Some rely on additional API's that are not available on all handsets.
- Because of these problems, many developers still end up developing their own GUI libraries from scratch, potentially delaying their projects for months. Our company faced the same problem when we started to develop our mobile applications. Since we were unable to find a suitable mobile GUI library, we decided to develop our own – the J2ME GUI library.

J2ME GUI

- Many different mobile devices, different screen sizes, colors , different input types
- Alerts
- Lists
- Form
- TextBox

Textbox

- allows the user to enter a String (title,name,password)
- depending on input may be

a tedious process

public TextBox(String title, String text, int maxSize, int constraints)

- title = screen title
- text = initial text on screen
- maxSize = maximum size of text box
- constraints – restrict input

TextBox - Constraints

- Constrain the input characters
  - TextField.ANY – allows any type of input supported by the device
  - TextField.NUMERIC– restricts to only integers
  - TextField.DECIMAL– allows numbers with fractional parts
  - TextField.PHONENUMBER – requires a telephone number
  - TextField.EMAILADDR – requires an email address
  - TextField.URL – requires

Text Box - Flags

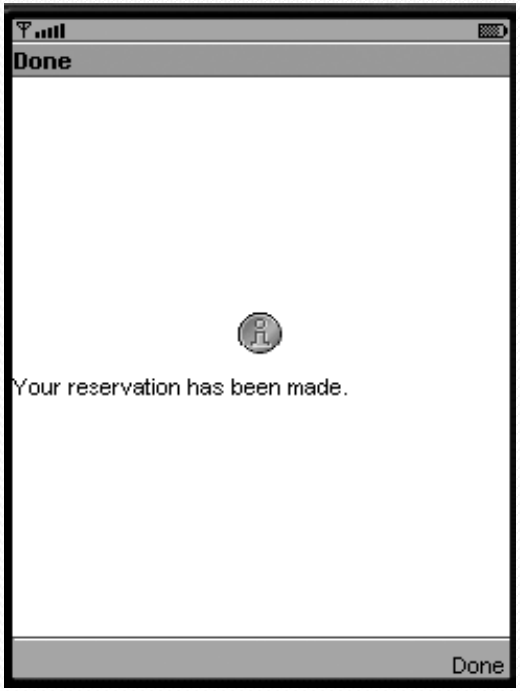
- TextField.PASSWORD
- TextField.UNEDITABLE
- TextField.SENSITIVE
- TextField.NON_PREDICTIVE
- TextField.INITIAL_CAPS_WORD
- TextField.INTIAL_CAPS_SENTENCE

- No Validation than use TextField.ANY and 0 for constraints parameter

Alerts

- alert types: ALARM, CONFIRMATION,ERROR,INFO, and WARNING

timed – certain amount of time – “Your transaction complete”
modal – until user dismisses it – “are you sure you want to quit?” “exit without saving?”



Alerts

public Alert() or

public Alert(String title, String alertText, Image alertImg, AlertType alertType)

- any or all parameters can be null

- default timeout , but can change timeout length
- Forever timeout means that it is modal

Alerts

- You can create an alert with:

Alert alt = new Alert("Sorry", "I Am sorry Dave", null, null);

- Set the timeout to 5 seconds by:

alt.setTimeout(5000);

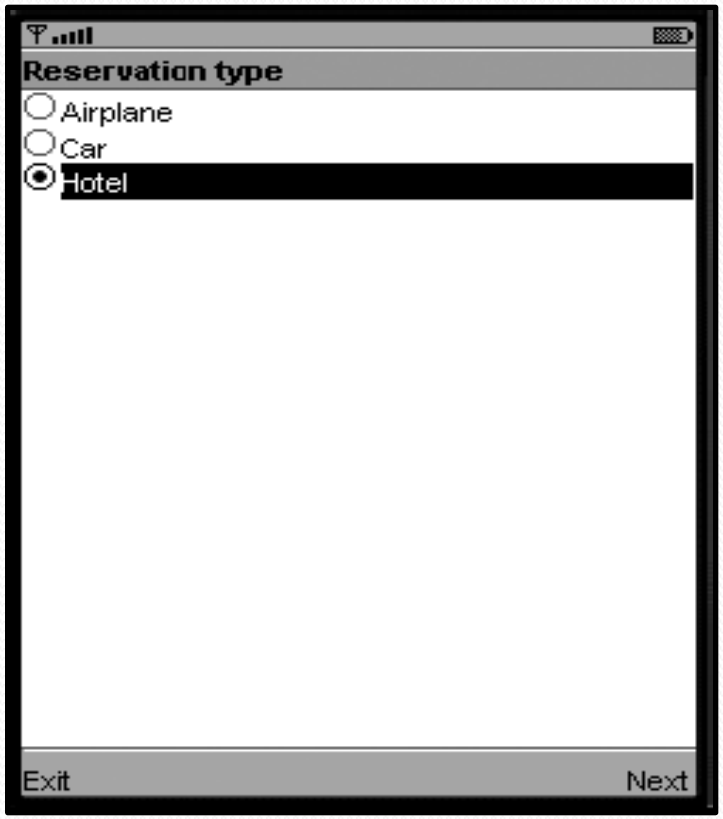
- Make it a modal alert by:

alt.setTimeout(FOREVER);

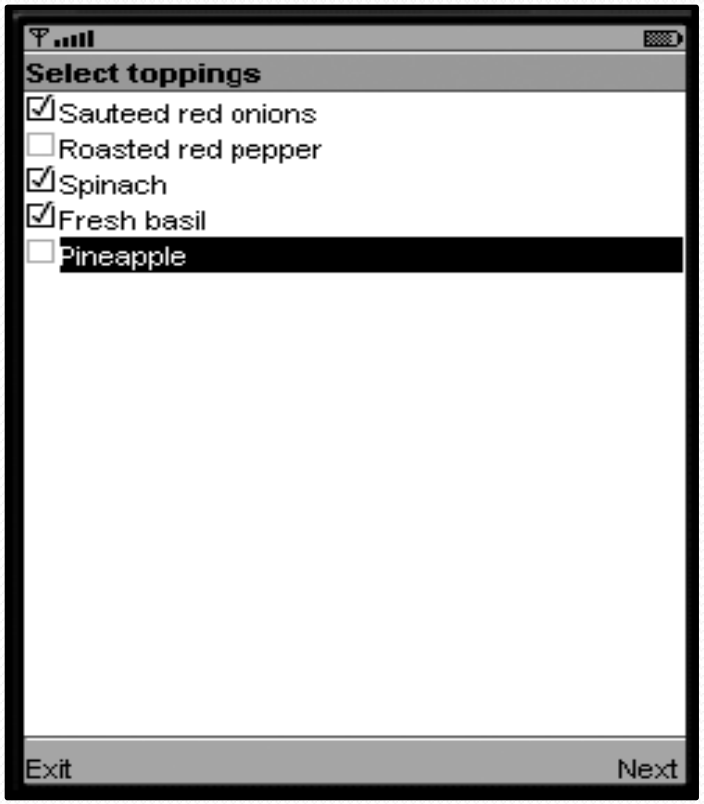
Lists

- users select items (called elements) from choices
- Exclusive -Single Selection – ex. radio buttons
- Multiple – Multiple Selection – ex. check list
- Text String or image is used to represent each element

Lists



Exclusive



Multiple

Creating Lists

public List(String title, int type)

public List(String title, int type,String[] stringElements,
Image[] imageElements)

Modifying Lists

- *public void set(int elementNum, String stringPart, Image imagePart)*
- *public void insert(int elementNum, String stringPart, Image imagePart)*
- *public int append(String stringPart, Image imagePart)*

Modifying Lists

- public String getString(int elementNum)
- public String getImage(int elementNum)
- public void delete(int elementNum)
- public void deleteAll()
- public boolean isSelected(int index)
- public int getSelectedIndex()
- public void setSelectedIndex(int index, boolean selected)

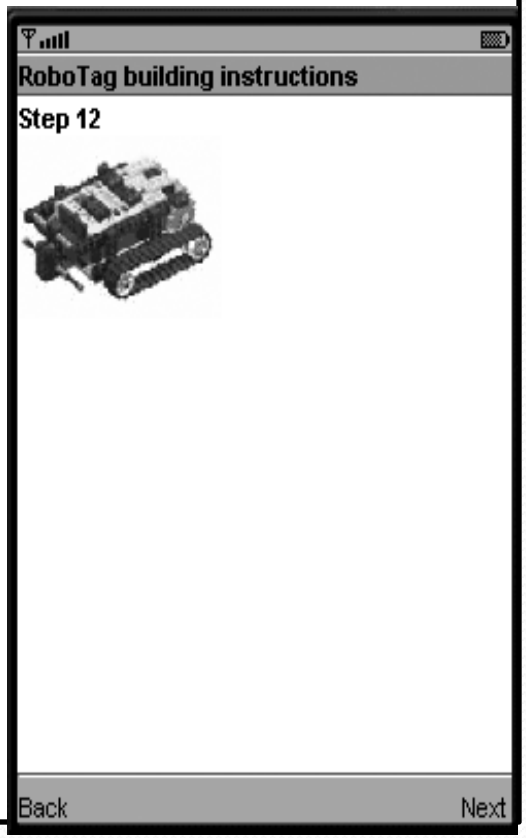
Forms

- A form includes collection of UI controls called Items

public Form(String title)

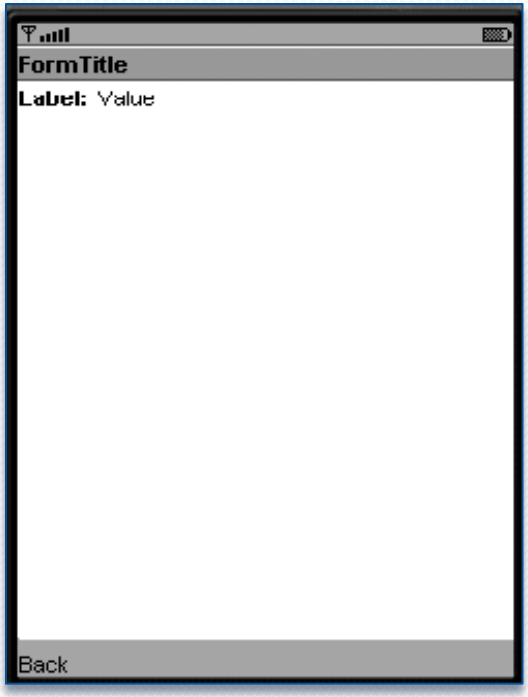
public Form(String title,Item[] items)

- public int append()
- public void set(int index,Item item)
- public void delete(int index)
- public void deleteAll()
- public int size()
- public Item get(int index)



Forms example

```
Form form = new Form("Form Title");
StringItem strItem = new StringItem("Label:",
 "Value");
form.append(strItem);
```



Forms - Items

- String , textfield, image Items, datefield
- Choice Group – similar to Lists before
- events and item changes as well
- Can create custom items to use on your own and now you can build up almost any type of UI component to make your needs

Form Layout

- Left to right, rows top to bottom
- items have labels, and can also have commands
- set size
- set layouts for individual items
  - setLayout(); getLayout();
  - LAYOUT_LEFT, LAYOUT_CENTER….

Commands

To Create a command, you need a name, type and also a priority.

Ex:
Command c = new Command("OK", Command.OK, 0);

public void addCommand(Command cmd)

public void removeCommand(Command cmd)

Command Types

There are different types of commands available for you to use:
Command.OK – Confirms a selction

Command.CANCEL – Cancels pending changes

Command.BACK – Moves the user back to a previous screen

Command.STOP – Stop a running operation

Command.HELP – Shows application Instructions

Command.SCREEN – indicates generic type for specific application commands

```
Command c = new Command("Launch", Command.SCREEN, 0);
```

GUI J2M features

- J2ME GUI tries to address all the pitfalls other GUI libraries suffer from by focusing on compatibility and performance without extravagant memory usage nor compromising aesthetics. The J2ME GUI library offers the following advantages:

- Visually appealing
- You can now develop appealing user interfaces for your mobile applications using the core components provided by J2ME GUI. Its default theme provides a professional look that users will be comfortable with.

GUI J2M features

- Easy porting of desktop applications
- Companies can port their desktop applications with minimal effort and in record time. All the typical components developers are accustomed to on desktop platforms are now available for the mobile platform. Their behavior is very similar to that of their desktop counterparts, with similar properties and functions, which greatly reduce the time it takes developers to learn how to use the library.
-

GUI J2M features

- Develop familiar intuitive interfaces
- Provide the usability your clients expect from your applications. With desktop-like interfaces, users will instantly find your applications comfortable to use.
- Fully customizable styling
- J2ME GUI's components can be styled to fit your application's needs, providing you with full control and flexibility. There are no prefixed colors in J2ME GUI and hundreds of style properties are exposed to control the finest details. The image glyphs used for certain components can also be replaced.
- Since styling code can comprise of several hundred lines, it would greatly affect your application's JAR size. Therefore, J2ME GUI features compiled style files that can be created and edited with the J2ME GUI Styler application included with the library. A typical style file of over 300 lines will only require 1Kb of the JAR file.

GUI J2M features

- Extend J2ME GUI with custom components
- You can easily extend any of the components in the J2ME GUI library, or even write a completely new component by extending the Component or Container classes. The graphics engine will automatically handle the rendering of your new components and treat them like any other library components. Additionally, we will also extend the library from time to time, adding new components or new properties and functions to existing ones.

GUI J2M features

- Superior performance
- Extensive research and development was conducted to acquire top performance whilst minimizing JAR size, as well as runtime memory. Our proprietary graphics engine is the product of aggressive tweaking techniques and optimized execution strategies. It ensures very high frame rates and instant, responsive interfaces even on low-end phones. In addition, large forms with many components have little impact on the achieved frame rate.

GUI J2M features

- Minimal footprint
- J2ME GUI features one of the smallest footprints for mobile GUI libraries. Resource files have been kept to a minimum and those that were included were optimized for size. Since J2ME GUI is able to render its own interpretations of components using vectors, all image glyphs can be removed to aid in size reduction. J2ME GUI library supports full obfuscation which can reduce its size by a further 30% to 40%. The resulting JAR file easily satisfies even the oldest mobile handsets with runtime memory consumption not much higher than the JAR file itself.

GUI J2M features

- Compatible with more than 90% of mobile handsets
- Research reveals that J2ME GUI is compatible with more than 90% of mobile phones currently in the market and this number continues to grow. Handset incompatibilities and bugs have been taken into consideration whilst developing the library and workarounds have been implemented where possible. Be careful of other products claiming compatibility with all handsets.
- Only the minimal set of functions was used in developing J2ME GUI, using the lowest level graphical instructions. It does not rely on any additional API's and should be able to run on nearly all MIDP 2.0 and CLDC 1.1 handsets.
- Now you can release your applications with greater confidence that they will run as expected on the widest range of handsets without requiring multiple builds.

GUI J2M features

- Community assistance
- Get assistance from co-developers via our developers' forum. Our own developers will also watch the forum when time permits to answer any questions. You can also use it to report any bugs or to communicate feature requests. The forum is open to everyone and we encourage all J2ME GUI developers to make use of the online forum.

Elements GUI

- All MIDP GUI classes are contained in the javax.microedition.lcdui package. This package contains three interfaces and twenty-one classes, as shown in Table 1 and Table 2.
- **Table 5-1: lcdui interfaces**
- Choice
- Defines an API for a user interface component that implements a selection from a predefined number of choices
- CommandListener
- Used by applications that need to receive high-level events from implementations
- ItemStateListener
- Used by applications that need to receive events that indicate changes in the internal state of the interactive items

Elements GUI

- **Table 2: lcdui classes**
- **Class**
- **Description**
- Alert
- A screen that shows data to the user and waits for a certain period of time before proceeding to the next screen.
- AlertType
- A utility class that indicates the nature of the alert.
- Canvas
- The base class for writing applications that need to handle low-level events and to issue graphics calls for drawing to the display.

Elements GUI

- ChoiceGroup
- A group of selectable elements intended to be placed within a Form.
- Command
- A construct that encapsulates the semantic information of an action.
- DateField
- An editable component for presenting calendar data and time information that may be placed into a Form.
- Display
- A utility that represents the manager of the display and input devices of the system.
- Displayable
- An object that has the capability of being placed on the display.
- Font
- A utility that represents font and font metrics.

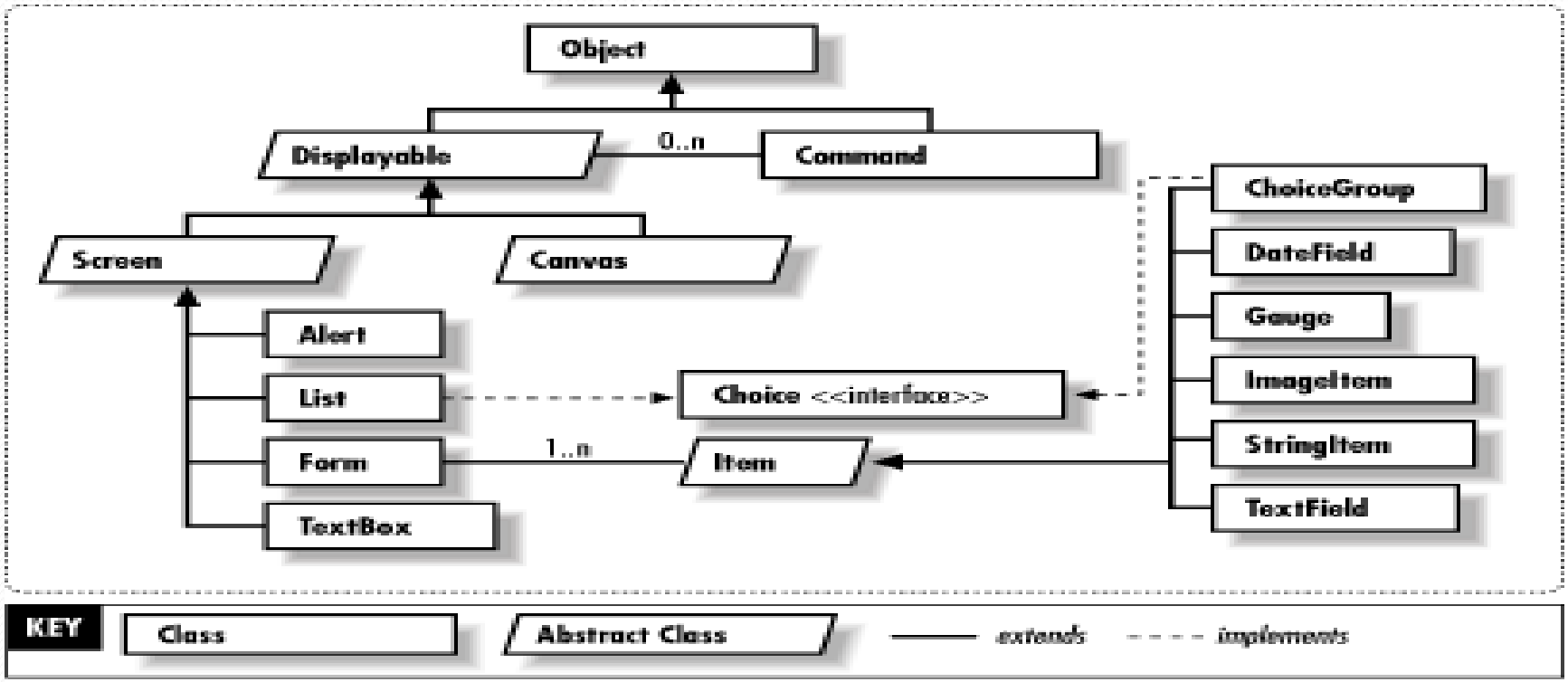
Elements GUI

- Form
- A screen that contains an arbitrary mixture of items (images, text, text fields, or choice groups, for instance).
- Gauge
- A utility that implements a bar graph display of a value intended for use in a form.
- Graphics
- A utility that provides a simple two-dimensional geometric rendering capability.

Elements GUI

- Image
- A utility that holds graphical image data.
- ImageItem
- A utility that provides layout control when Image objects are added to a form or alert.
- Item
- A superclass for all components that can be added to a Form or Alert.
-
- List
- A screen containing a list of choices.
- Screen
- The superclass of all high-level user interface classes.
- StringItem
- An item that can contain a String.
- TextBox
- A screen that allows the user to enter and edit text.
- TextField
- An editable text component that can be placed into a Form.
- Ticker
- A ticker-type piece of text that runs continuously across the display. It can be attached to all screen types except Canvas.

shows the major classes and the relationships between them.



example

- **Example**
- This example demonstrates how to create various GUI components. The MIDlet for this example allows you to test lists, forms, choices, gauges, text fields, text boxes, for instance.
- The GuiTests MIDlet has a few methods for testing various GUI components. The MIDlet makes use of the following classes, listed in alphabetical order, from the javax.microedition.lcdui package:
- Alert
- Command
- DateField
- Display
- Displayable
- Form
- Gauge
- List
- TextBox
- TextField
- Ticker
- CommandListener

Advantages and Disadvantages of GUIs

ADVANTAGES

On average, users can readily operate the computer and feels more in control and not intimidated by it

User learning time is short – does not take long to open the box, assemble the kit and start to do productive work

With prompts and alerts and mouse movements and double-clicking on objects, etc. users get instant feedback on their actions

Mistakes can be more readily detected than with command line interfaces and they can be more easily corrected

Once you have learned to use one WIMP based operating system then it is simpler to transfer those skills to using a different OS eg Windows VS Mac OS

Advantages and Disadvantages of GUIs

DISADVANTAGES

Graphical user interfaces need large amounts of memory and fast processors to display the images and manipulate them interactively

Skilled users feel that they can accomplish tasks using fewer steps than with a GUI eg copying a file – CP <oldfilename>, <newfilename>

Users with certain disabilities (vision/motor) can still have trouble using GUIs

Screens can easily become cluttered and difficult to navigate

References

- http://www.onjava.com/pub/a/onjava/excerpt/wirelessjava_ch5/index1.html
- http://developers.sun.com/mobility/midp/articles/ui/ - Sun's J2ME GUI tutorial
- **http://www.j2mepolish.org/ - A GUI Designer which may be useful for your own projects beyond this course**

References

- http://jcp.org/jsr/detail/30.jsp
- http://java.sun.com/products/consumer-embedded/
- http://java.sun.com/j2me/j2me-ds.pdf
- http://wireless.java.sun.com/
- http://www.scc-kk.co.jp/lib_scc/catalog/books/B-228/B-228.pdf
- http://chinaunix.net/jh/26/128217.html
- http://developer.java.sun.com/developer/products/j2me/
- http://wireless.java.sun.com/midp/articles/#appmodels
- http://wireless.java.sun.com/midp/articles/getstart/
- http://www.java.com/en/learn/mobile

J2ME
Java Mobility Software

شكراً جزيلاً