QuizActivity

```
onCreate (Budle instanceState)– called by OS
        trueButton ->   checkAnswer
        falseButton->   checkAnswer
        nextButton->    increment question id
                        reset isCheater variable, updateQuestion
        cheatButton ->
                        COMMUNICATE BETWEEN ACTIVITIES WITH INTENTS
                        get the expected answer
                        Create intent with this QuizActivity as a context and the CheatActivity.class
                                Add the expected answer to the intent with the URL.answer_is_true key
                                Explain CheatActivity.this
                        *startActivityForResult(intent, MY_ID)
                                explain My_ID – saved in OS and returned in onActivityResult
        If the input bundle exists get the current question id from the bundle
        updateQuestion

checkAnswer
        get the expected answer
        if the user cheated -> get id for the cheat response string – mIsCheater is a field
        if the user's answer matched the expected answer -> get the id for the correct response
        if the user's answer does not match the expected answer -> get the id for the incorrect response
        Create the toast

updateQuestion
        using the current index get the id of question
        *use the id to change the displayed text in the Question view

*onSaveInstanceState – called by OS
        store question id into bundle with key

*onActivityResult – called by OS if we started the activity asking for a result(startActivityForResult)
        (requestCode – predefined code (in Activity) indicating success of called activity
         resultCode – is this a result from starting and running the the Activity from this activity
        Intent – any information to be passed back)

        If the activity ran correctly and this is a response from my call starting the activity and there is an intent
                Get the boolean value indicating whether the user cheated or not from the intent
                If there was no result assume the user did not cheat
```

Cheat Activity

```
onCreate(Bundle instanceState)
    get the expected answer from the intent
        *getIntent()
    get the Answer View (which is initially blank)
    get the Show Answer Button ->    show the expected answer in the AnswerView
                                     setAnswerShownResult(true)

setAnswerShownResult(isAnswerShown -- which is always true)
    Create the intent to send to previous activity (we don't use the one we received)
    Put the indicator as to whether we cheated or not – (if we got here, we did cheat)
    *setResult(request code – RESULT_OK inherited from Activity, the new return intent)
        setResult is also inherited from activity

    the information saved by calling setResult will be used in the Quiz activity the OS calls
    onActivityResult
```

Fragments

Look at shippingFrameLayout and billingFrameLayout in the xml for activityOrderInfo

MainActivity

```
onCreate(Bundle savedIndstanceState)
    get the only button -> onButtonClicked()

onButtonClicked()
    create new intent from (this, OrderInfoActivity.class);
    startActivityForResult(intent, this activities id)
        we are really starting a fragment

onActivityResult(requestCode, resultCode, return intent)
    if this method was called in response to me starting the OrderInfoActivity and the result code shows success and there is an intent
        Get the order from the intent and log it.
```

OrderInfoActivity is a FragmentActivity (an Activity that knows how to manage fragments)

```
    Domain: see xml for layout
        shippingAddressFragment
        billingAddressFragment
        checkbox
        button

    onCreate(Bundle savedInstanceState) – called by OS
        getCheckBox
            make is checked
            setOnCheckChangeListener(OnCheckedChangeListener)
                override onCheckedChanged(CompoundButton, boolean isChecked) to call
                    onCheckChanged
        getButton ->onButtonClicked()

        -- create the shippingAddressFragment as instance of AddressFragment
        *get the FragmentManager  fm – this.getSupportFragmentManager()
                defined in FragmentActivity
            fm.getFragmentById (R.id.shippingFrameLayout -- defined in xml)
                get the fragment associated with the FrameLayout, if one exists
            cast to AddressFragment(user defined)
            If it has not been defined
                Create new shippingAddressFragment from  AddressFragment.newInstance
                    –method is  user defined in AddressFragment
                    Passing in Header String
            fm.beginTransaction.add(id of the shippingFrameLayout, shippingAddressFragment)
                then commit it
                -- adding the fragment to the GUI

    *onCheckChanged () called when checkbox checked or unchecked
        get the fragment manager
        if the checkbox (a field in this class) is checked (just a moment ago)
            remove the billingAddressFragment from the GUI
            fm.beginTransactin().remove(billingAddressFragment).commit()
        else if the checkbox Is not checked – which occurred just a moment ago
            create the new billingAddressFragment – as above
                AddressFragment.newInstance(headerString)
                Fm.beginTransaction().add(billingAddressFragment).commit()
                -- add the billingAddress to the GUI

    onButtonClicked()
        returnResult() – user defined, setResult then kill this activity
        this.finish() – stop this activity
```

- returnResult() – not really returned but setResult
  - get the shipping address from the shippingAddress fragment (Address is a user defined class)
  - if the billing address exists – get it from the billing addressFragment
  - create an Order and add the shipping address and billing address
  - create return intent
    - add the order to it
  - setResult(RESULT_OK, returnIntent) – saving info for the onActivityResult method in the parent

AddressFragment – looks a lot like an activity

- Look at XML
  - Title:TextView
    - textAppearance
  - GridLayout: 4 rows, 2 columns
  - Each row has a text view (a label) and a EditText field – text you can type in
- Domain:
  - title: String
  - TitleView : TextView
  - StreetAddress : EditText
  - City:EditText
  - State:EditText
  - ZipCode:EditText

- static newInstance(title) – creating a new fragment, called by Order InfoActivity
  - **create a new instance of this class – a new fragment**
  - **create a new bundle**
  - **put the title in the bundle**
  - **put the bundle in the fragment**
  - **return this fragment**

- onCreate(Bundle savedInstanceState) – called by the OS
  - *__getArguments() -- get the arguments saved when this fragment was created__
    - **– only defined for fragments**
    - **-- see “newInstance” above**
  - if there are arguments then get the title from it and store it in the field “title”

```
*onCreateView(LayoutInflater, ViewGroup, Bundle – savedInstanceState) – called from OS
        View v = inflater.inflate(R.layout.fragment_address, -- the xml file name
                                container – views new parent,
                                boolean addToParentNow)
        get the textView and set it to the string in the field "title"
        initialize all of the textViews
        return the newly created view

getAddress() – called from the returnResult() method in OrderInfoActivity, called when button clicked
        -- called twice, once for the shipping address and if it exists once for the billing address
        create empty instance of the Address class
        set each of the street, city, state, and zip code from the respective TextViews
                xTextView.getText().toString()
```