1. How to create a "menu" in Android, and what are the different items of the "menu"?
To create a menu in Android, lets define a menu XML file in the `res/menu/` directory. This file contains `<item>` elements, each representing a menu item. We can also use <menu> elements to create submenus. Menus can be of different types: Options Menu, Context Menu, and Popup Menu.

2. What are the commonly used `<item>` attributes in Android applications for "menu"?
Commonly used `<item>` attributes include:
`android:id`: Uniquely identifies the menu item.
`android:icon`: Sets the item's icon from the drawable folder.
`android:title`: Sets the item's text.
`android:showAsAction`: Specifies how the item appears in the app bar (e.g., `ifRoom`, `always`, `never`).

3. How to call the method "MenuInflater.inflate()" for "menu"? Write the code.
calling `MenuInflater.inflate()` within the `onCreateOptionsMenu` method of the activity or fragment.Code:

```
@Override
public void onCreateOptionsMenu(Menu menu, MenuInflater inflater)
{
    inflater.inflate(R.menu.menu_example, menu);
    super.onCreateOptionsMenu(menu, inflater);
}
```

4. What is the method to handle a menu item click events? Write the code.
To handle menu item click events, lets implement the `onOptionsItemSelected` method in the activity or fragment.

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {
        case R.id.mail:
            // Handle the "mail" menu item click
            return true;
        case R.id.share:
            // Handle the "share" menu item click
            return true;
        default:
            return super.onOptionsItemSelected(item);
    }
}
```

5. Why to use onContextItemSelected()?
`onContextItemSelected()` is used to handle context menu item click events. Context menus are typically used when we want to provide options related to a specific context, such as long-pressing on an item in a list or a view. By using `onContextItemSelected()`, we can respond to user actions that affect the selected content or context frame, providing a more intuitive user experience in such scenarios.

6. What are the different Options Menu Attributes in Android?
The Options Menu in Android typically consists of the following attributes:
- <menu>: Root node that contains one or more `<item>` elements.
- <item>: Represents menu items.
- android:id: Uniquely identifies elements in the application.
- android:icon: Sets the item's icon from the drawable folder.
- android:title: Sets the item's title.
- android:showAsAction: Specifies how the item should appear as an action item in the app bar.

7. Which method is used to activate/override the option menu (onCreateOptionsMenu()) ?
The onCreateOptionsMenu(Menu menu) method is used to activate/override the Options Menu in an Android activity or fragment. Developers override this method to define the menu items that should appear in the Options Menu.

8. Which method is used to load the menu resource (MenuInflater.inflate()) ?
The MenuInflater.inflate(int menuRes, Menu menu) method is used to load the menu resource from an XML file into the Options Menu. This method is typically called within the `onCreateOptionsMenu()` method to populate the menu with items defined in an XML resource.

9. Which method is used to handle item click events for the option menu (onOptionsItemSelected()) ?
The onOptionsItemSelected(MenuItem item) method is used to handle item click events for the Options Menu. Developers override this method to perform specific actions when a menu item is clicked. The `MenuItem` parameter provides information about which item was clicked.

10. Write the purpose of using `getMenuInflater()` in Android.
`getMenuInflater()` is a method provided by Android to obtain a MenuInflater object. The `MenuInflater` is used to inflate (parse and load) menu XML resources into a Menu object. It is typically used in the `onCreateOptionsMenu()` method of an activity or fragment to populate the Options Menu with items defined in an XML resource. In the code examples we provided, getMenuInflater() is used to inflate the menu XML resource and add menu items to the Options Menu.

11. How to use the `registerForContextMenu()` method for registering a view for a context menu?**

To register a view for a context menu, **we** can use the `registerForContextMenu(View view)` method in our activity or fragment. This method specifies which view will trigger the context menu when it's long-pressed or clicked. For example:

```java
Button btn = (Button) findViewById(R.id.btnShow);
registerForContextMenu(btn);
```

12. Which method is used to handle item click events for a context menu (`onContextItemSelected()`)?**

To handle item click events for a context menu, **we** can override the `onContextItemSelected(MenuItem item)` method in our activity or fragment. This method is called when a context menu item is selected. Here's an example:

```java
@Override
public boolean onContextItemSelected(MenuItem item) {
    if (item.getTitle().equals("Upload")) {
        // Handle the "Upload" menu item click
    } else {
        return false;
    }
    return true;
}
```

13. Difference between context menu and option menu in Android:**

- **Context Menu:**

- Appears when **we** long-press or click on a specific view or element.
- Used for actions that affect the selected content or context.
- Typically used for context-specific actions.
- Doesn't support item shortcuts or item icons.

- **Options Menu:**
- Appears when **we** press the device's "Menu" button or, in modern Android versions, as a three-dot icon in the app bar.
- Contains actions that have a global impact on the app.
- Usually provides actions like settings, search, and other global options.
- Supports item shortcuts and item icons.

14. Write the `showPopup()` method code.**

To display a Popup Menu in Android, we can create a method like `showPopup()` to show the Popup Menu when a button is clicked. Here's the code for the `showPopup()` method:

```java
public void showPopup(View v) {
    PopupMenu popup = new PopupMenu(this, v);
    popup.setOnMenuItemClickListener(this);
    popup.inflate(R.menu.popup_menu);
    popup.show();
}
```

In this method, we:
- Create a `PopupMenu` instance anchored to the provided `View` (in this case, `v`).
- Set an `OnMenuItemClickListener` to handle menu item clicks.
- Inflate the menu resource (`R.menu.popup_menu`) to populate the Popup Menu.
- Finally, call `popup.show()` to display the Popup Menu.

15. Which method is used to handle item click event for a popup menu (`onMenuItemClick()`)?**

To handle item click events for a Popup Menu in Android, we can implement the `PopupMenu.OnMenuItemClickListener` interface and override its `onMenuItemClick(MenuItem item)` method. Here's an example:

```java
@Override
public boolean onMenuItemClick(MenuItem item) {
    switch (item.getItemId()) {
        case R.id.search_item:
            // Handle the "Search" menu item click
            return true;
        case R.id.upload_item:
            // Handle the "Upload" menu item click
            return true;
        case R.id.copy_item:
            // Handle the "Copy" menu item click
            return true;
        case R.id.print_item:
            // Handle the "Print" menu item click
            return true;
        case R.id.share_item:
            // Handle the "Share" menu item click
            return true;
```

```
        case R.id.bookmark_item:
            // Handle the "BookMark" menu item click
            return true;
        default:
            return false;
    }
}
```

In this method, we check the `item.getItemId()` to determine which menu item was clicked and then perform the corresponding action.