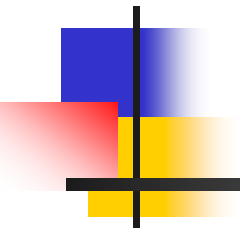
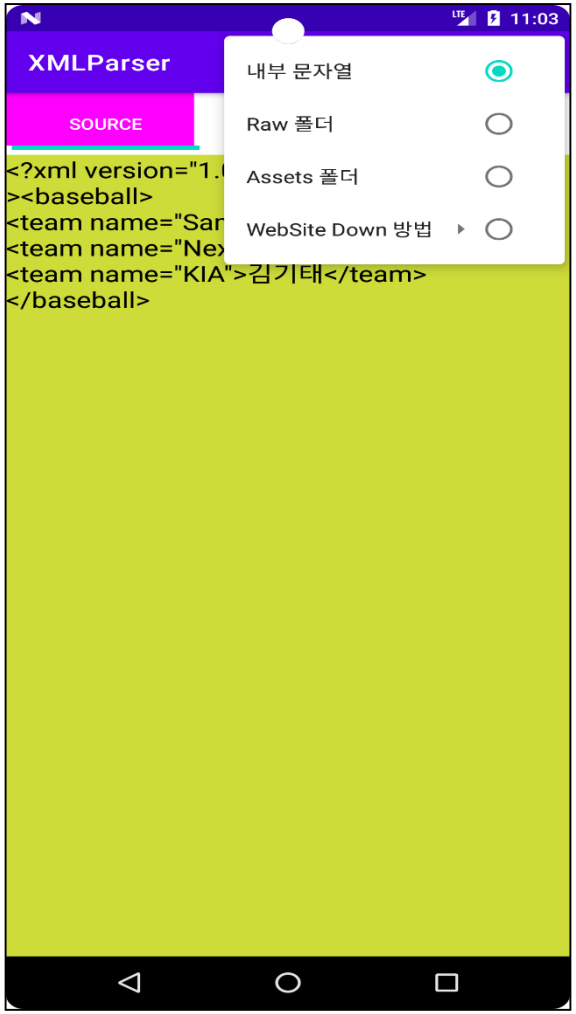
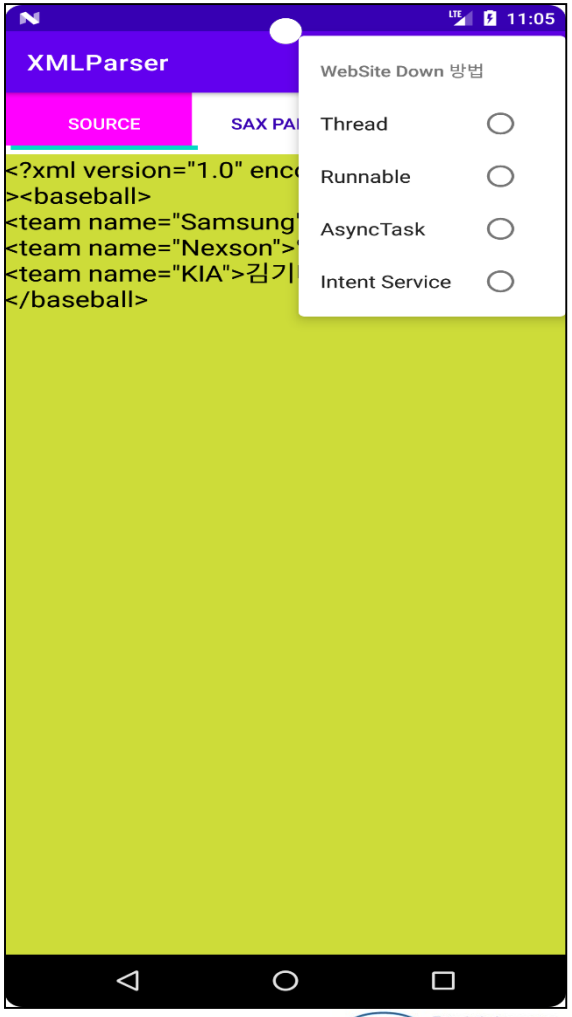


XML Parser 실습
(SAX Parser, PULL Parser)

배 희호 교수
경복대학교
소프트웨어융합과

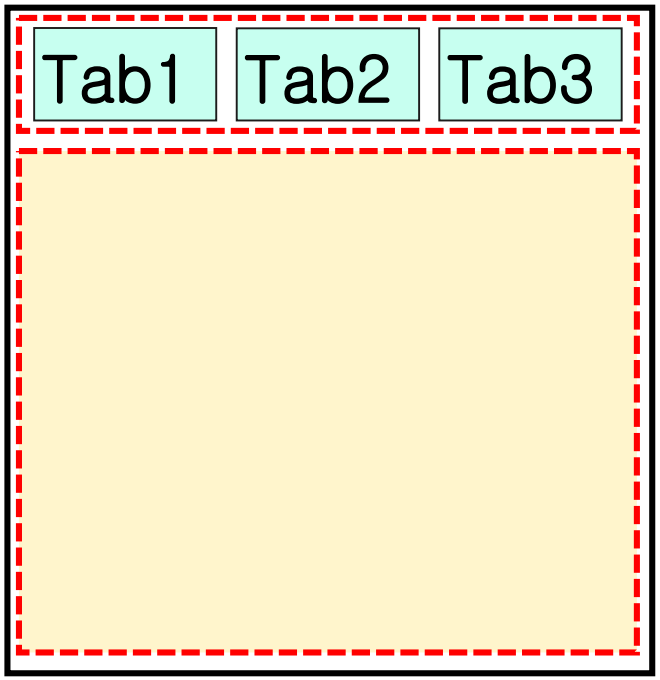
XML Parsing





XML Parsing

- TabLayout을 사용하는 방법
  - TabLayout은 단순한 Tab Interface의 경우 TabLayout만을 사용하여 구현할 수 있음
  - Tab을 선택할 때마다 FrameLayout의 Contents를 변경하여 각 Tab에 맞는 정보를 표시할 수 있음



TabLayout

- ✓ FrameLayout (fragment)
- ✓ ViewPager
- ✓ ViewPager2

XML Parsing

- 사용자 인터페이스

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity">

    <com.google.android.material.tabs.TabLayout
        android:id="@+id/tabs"
        style="@style/CustomTabLayout"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:elevation="1dp"
        android:padding="4dp"
        app:tabGravity="fill"
        app:tabMode="fixed" />
```

XML Parsing

- 사용자 인터페이스

```
    <androidx.viewpager2.widget.ViewPager2
        android:id="@+id/viewPager"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        app:layout_behavior="@string/appbar_scrolling_view_behavior" />
</LinearLayout>
```

XML Parsing

- menu.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <group android:checkableBehavior="single">
        <item
            android:id="@+id/item1"
            android:checked="true"
            android:title="내부 문자열" />
        <item
            android:id="@+id/item2"
            android:title="Raw 폴더" />
        <item
            android:id="@+id/item3"
            android:title="Assets 폴더" />
        <item android:title="WebSite">
```

XML Parsing

- menu.xml

```
<menu>
    <group android:checkableBehavior="single">
        <item
            android:id="@+id/item4"
            android:title="Thread" />
        <item
            android:id="@+id/item5"
            android:title="Runnable" />
        <item
            android:id="@+id/item6"
            android:title="AsyncTask" />
        <item
            android:id="@+id/item7"
            android:title="Intent Service" />
    </group>
</menu>
</item>
</group>
</menu>
```

XML Parsing

- MainActivity.JAVA

```
public class MainActivity4 extends AppCompatActivity {
    private int type = R.id.item1;
    private ViewPager2 viewPager;
    private FragmentAdapter adapter;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main4);

        TabLayout tabs = findViewById(R.id.tabs);
        viewPager = findViewById(R.id.viewPager);
        adapter = new FragmentAdapter(this);
        update();
```

XML Parsing

- MainActivity.JAVA

```
TabLayoutMediator mediator = new TabLayoutMediator(tabs, viewPager,
                        new TabLayoutMediator.TabConfigurationStrategy() {
    @Override
    public void onConfigureTab(@NonNull TabLayout.Tab tab, int position) {
        switch (position) {
            case 0:
                tab.setText("Source");
                break;
            case 1:
                tab.setText("SAX");
                break;
            case 2:
                tab.setText("PULL");
        }
    }
});
mediator.attach();
}
```

XML Parsing

- MainActivity.JAVA

```
protected void update() {
    SourceFragment sourceFragment = new SourceFragment(this, type);
    PULLFragment pullFragment = new PULLFragment(this, type);
    SAXFragment saxFragment = new SAXFragment(this, type);
    adapter.setFragments(sourceFragment, saxFragment, pullFragment);
    viewPager.setAdapter(adapter);
}
```

XML Parsing

- MainActivity.JAVA

```
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        MenuInflater inflater = getMenuInflater();
        inflater.inflate(R.menu.menu, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        type = item.getItemId();
        item.setChecked(true);
        update();
        return true;
    }
}
```

XML Parsing

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#CDDC39"
    android:orientation="vertical"
    tools:context=".MainActivity">

    <ScrollView
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

        <TextView
            android:id="@+id/textView"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:textSize="20dp" />
    </ScrollView>
</LinearLayout>
```

XML Parsing

- FragmentAdapter.JAVA

```
public class FragmentAdapter extends FragmentStateAdapter {
    private SourceFragment sourceFragment;
    private SAXFragment saxFragment;
    private PULLFragment pullFragment;

    public FragmentAdapter(@NonNull FragmentActivity fragmentActivity) {
        super(fragmentActivity);
    }

    @NonNull
    @Override
    public Fragment createFragment(int position) {
        if (position == 0)
            return sourceFragment;
        else if (position == 1)
            return saxFragment;
        else
            return pullFragment;
    }
```

XML Parsing

- FragmentAdapter.JAVA

```
    @Override
    public int getItemCount() {
        return 3;
    }

    public void setFragments(Fragment sourceFragment,
                             Fragment saxFragment, Fragment pullFragment) {
        this.sourceFragment = (SourceFragment) sourceFragment;
        this.saxFragment = (SAXFragment) saxFragment;
        this.pullFragment = (PULLFragment) pullFragment;
        notifyItemChanged(0);
    }
}
```

XML Parsing

- SourceFragment.JAVA

```
public class SourceFragment extends Fragment {
    private MyApplication app;
    private Activity activity;
    private int type;
    private TextView textView;

    public SourceFragment(Activity activity, int type) {
        this.activity = activity;
        this.type = type;
        app = (MyApplication) activity.getApplication();
    }
```

XML Parsing

- SourceFragment.JAVA

```
private Handler handler = new Handler(Looper.getMainLooper()) {
    public void handleMessage(Message message) {
        if (message.what == 1) {
            app.setXml((String) message.obj);
            textView.setText(app.getXml());
        } else
            Toast.makeText(activity, "다운로드 실패함", Toast.LENGTH_LONG).show();
    }
};

@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
                                                    Bundle savedInstanceState) {
    ViewGroup rootView = (ViewGroup) inflater.inflate(R.layout.source_fragment,
                                                    container, false);
    textView = rootView.findViewById(R.id.textView);
    return rootView;
}
```

XML Parsing

- SourceFragment.JAVA

```
@Override
public void onViewCreated(@NonNull View view,
                          @Nullable Bundle savedInstanceState) {
    if (type == R.id.item1) {
        DOMMaker domMaker = new DOMMaker(activity);
        app.setXml(domMaker.source());
    } else if (type == R.id.item2) {
        ReadRaw raw = new ReadRaw(activity);
        app.setXml(raw.get(R.raw.order));
    } else if (type == R.id.item3) {
        ReadAssets assets = new ReadAssets(activity);
        app.setXml(assets.get("student.xml"));
    } else if (type == R.id.item4) {
```

XML Parsing

- SourceFragment.JAVA

```
    DownloadThread thread = new DownloadThread(activity, app.getPage(0));
    thread.start();
    try {
        thread.join();
        app.setXml(thread.getResult());
    } catch (InterruptedException e)
        Toast.makeText(activity, e.getMessage(), Toast.LENGTH_SHORT).show();
} else if (type == R.id.item5) {
    DownloadRunnable runnable =
                            new DownloadRunnable(activity, app.getPage(1));
    Thread thread = new Thread(runnable);
    thread.start();
    try {
        thread.join();
        app.setXml(runnable.getResult());
    } catch (InterruptedException e)
        Toast.makeText(activity, e.getMessage(), Toast.LENGTH_SHORT).show();
} else if (type == R.id.item6) {
```

XML Parsing

- SourceFragment.JAVA

```
            DownloadAsyncTask task = new DownloadAsyncTask(activity);
            try {
                app.setXml(task.execute(app.getPage(2)).get());
            } catch (ExecutionException | InterruptedException e) {
                Toast.makeText(activity, e.getMessage(),
                        Toast.LENGTH_SHORT).show();
            }
        } else if (type == R.id.item7) {
            Intent intent = new Intent(activity, DownloadService.class);
            Messenger messenger = new Messenger(handler);
            intent.putExtra("Messenger", messenger);
            intent.putExtra("page", app.getPage(3));
            activity.startService(intent);
        }
        textView.setText(app.getXml());
        textView.setTextColor(Color.BLACK);
    }
}
```

XML Parser

- DOMMaker.JAVA

```
public class DOMMaker {
    private Context context;

    public DOMMaker(Context context) {
        this.context = context;
    }

    public String source() {
        StringWriter writer = null;
        String[][] team = {{"Samsung", "류중일"}, {"Nexson", "엽경엽"},
                                                    {"KIA", "김기태"}};
        try {
            DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
            DocumentBuilder builder = factory.newDocumentBuilder();
            Document document = builder.newDocument();

            Element root = document.createElement("baseball");
            document.appendChild(root);
```

XML Parser

```
            for (int i = 0; i < team.length; i++) {
                Element element = document.createElement("team");
                element.setAttribute("name", team[i][0]);
                element.appendChild(document.createTextNode(team[i][1]));
                root.appendChild(element);
            }
            TransformerFactory factory1 = TransformerFactory.newInstance();
            Transformer former = factory1.newTransformer();
            former.setOutputProperty(OutputKeys.OMIT_XML_DECLARATION, "no");
            former.setOutputProperty(OutputKeys.INDENT, "yes");
            writer = new StringWriter();
            StreamResult result = new StreamResult(writer);
            DOMSource source = new DOMSource(document);
            former.transform(source, result);
        } catch (ParserConfigurationException | TransformerException e) {
            Toast.makeText(context, "", Toast.LENGTH_SHORT).show();
        }
        return String.valueOf(writer);
    }
}
```

XML Parser

- DownloadThread.JAVA

```
public class DownloadThread extends Thread{
    private Context context;
    private String page;
    private StringBuilder builder;
    private Handler handler;

    public DownloadThread(Context context, String page) {
        this.context = context;
        this.page = page;
        builder = new StringBuilder();
        handler = new Handler();
    }
```

XML Parser

- DownloadThread.JAVA

```
@Override
public void run() {
    try {
        URL url = new URL(page);
        InputStream inputStream = url.openStream();
        InputStreamReader streamReader =
                                new InputStreamReader(inputStream, "UTF-8");
        int ch;
        while ((ch = streamReader.read()) != -1) {
            builder.append((char) ch);
        }
        streamReader.close();
        inputStream.close();
    } catch (final IOException e) {
```

XML Parser

- DownloadThread.JAVA

```
            handler.post(new Runnable() {
                @Override
                public void run() {
                    Toast.makeText(context, e.getMessage(),
                                                        Toast.LENGTH_SHORT).show();
                }
            });
        }
    }

    public String getResult() {
        return builder.toString();
    }
}
```

XML Parser

- DownloadRunnable.JAVA

```
public class DownloadRunnable implements Runnable{
    private Context context;
    private String page;
    private StringBuilder builder;
    private Handler handler;

    public DownloadRunnable(Context context, String page) {
        this.context = context;
        this.page = page;
        builder = new StringBuilder();
        handler = new Handler();
    }
```

XML Parser

- DownloadRunnable.JAVA

```
@Override
public void run() {
    try {
        URL url = new URL(page);
        URLConnection connection = url.openConnection();
        InputStream inputStream = connection.getInputStream();
        InputStreamReader streamReader =
                                    new InputStreamReader(inputStream, "UTF-8");
        BufferedReader reader = new BufferedReader(streamReader);
        String line;
        while ((line = reader.readLine()) != null) {
            builder.append(line +'₩n');
        }
        reader.close();
        streamReader.close();
        inputStream.close();
    } catch (IOException e) {
```

XML Parser

- DownloadRunnable.JAVA

```
            handler.post(new Runnable() {
                @Override
                public void run() {
                    Toast.makeText(context, e.getMessage(),
                                                        Toast.LENGTH_SHORT).show();
                }
            });
        }
    }

    public String getResult() {
        return builder.toString();
    }
}
```

XML Parser

- DownloadAsyncTask.JAVA

```
public class DownloadAsyncTask extends AsyncTask<String, String, String> {
    private Context context;

    public DownloadAsyncTask(Context context) {
        this.context = context;
    }

    @Override
    protected String doInBackground(String... strings) {
        StringBuilder builder = new StringBuilder();
        try {
            URL url = new URL(strings[0]);
            HttpURLConnection connection =
                                    (HttpURLConnection) url.openConnection();
            InputStream inputStream = connection.getInputStream();
            InputStreamReader streamReader =
                                        new InputStreamReader(inputStream, "UTF-8");
```

XML Parser

- DownloadAsyncTask.JAVA

```
            Scanner scanner = new Scanner(streamReader);
            while (scanner.hasNext()) {
                builder.append(scanner.nextLine() +'₩n');
            }
            scanner.close();
            streamReader.close();
            inputStream.close();
            connection.disconnect();
        } catch (IOException e) {
            publishProgress(e.getMessage());
        }
        return builder.toString();
    }

    @Override
    protected void onProgressUpdate(String... values) {
        Toast.makeText(context, values[0], Toast.LENGTH_SHORT).show();
    }
}
```

XML Parser

- DownloadService.JAVA

```
public class DownloadService extends IntentService {
    private Handler handler;
    private String result;

    public DownloadService() {
        super("DownloadService");
        handler = new Handler();
    }
```

XML Parser

- DownloadService.JAVA

```
@Override
protected void onHandleIntent(Intent intent) {
    if (intent == null)
        return;
    Messenger messenger = intent.getParcelableExtra("Messenger");
    String page = intent.getStringExtra("page");
    DownloadRunnable downThread = new DownloadRunnable(this, page);
    Thread thread = new Thread(downThread);
    thread.start();
    try {
        thread.join();
        result = downThread.getResult();
    } catch (InterruptedException e) {
        Toast.makeText(this, e.getMessage(), Toast.LENGTH_SHORT).show();
    }
```

XML Parser

```
    thread = new Thread(new Runnable() {
        @Override
        public void run() {
            Message message = Message.obtain();
            message.what = 1;  // 메시지 타입
            message.obj = result;
            try {
                messenger.send(message);
            } catch (RemoteException e) {
                handler.post(new Runnable() {
                    @Override
                    public void run() {
                        Toast.makeText(getBaseContext(), e.getMessage(),
                                                        Toast.LENGTH_SHORT).show();
                    }
                });
            }
        }
    });
    thread.start();
}
```

XML Parser

- DownloadService.JAVA

```
    @Override
    public void onDestroy() {
        //   Toast.makeText(this, "Music Service가 중지되었습니다",
                                                    Toast.LENGTH_LONG).show();
    }
}
```

XML Parser

- SAXFragment.JAVA

```
public class SAXFragment extends Fragment {
    private MyApplication app;
    private Activity activity;
    private int type;
    private ListView listView;
    private TextView textView;

    public SAXFragment(Activity activity, int type) {
        this.activity = activity;
        this.type = type;
        app = (MyApplication) activity.getApplication();
    }
```

XML Parser

- SAXFragment.JAVA

```
@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
                                                  Bundle savedInstanceState) {
    ViewGroup rootView;
    if (type == R.id.item6 || type == R.id.item7) {
        rootView = (ViewGroup) inflater.inflate(R.layout.fragment_parse2,
                                                        container, false);
        listView = rootView.findViewById(R.id.listView);
    } else {
        rootView = (ViewGroup) inflater.inflate(R.layout.fragment_parse1,
                                                          container, false);
        textView = rootView.findViewById(R.id.textView);
    }

    return rootView;
}
```

XML Parser

- SAXFragment.JAVA

```
@Override
public void onViewCreated(@NonNull View view,
                                        @Nullable Bundle savedInstanceState) {
    MySAXParser parser = new MySAXParser(activity);
    if (type == R.id.item6) {
        ArrayList<Country> countries = parser.parsing6(app.getXml());
        ArrayAdapter<Country> adapter = new CountryAdapter(activity,
                R.layout.country, countries);
        listView.setAdapter(adapter);
    } else if (type == R.id.item7) {
        ArrayList<HashMap<String, String>> items = parser.parsing7(app.getXml());
        SimpleAdapter adapter = new SimpleAdapter(activity, items,
                R.layout.list_item,
                new String[]{"ITEM", "제조사", "MODEL", "COST"},
                new int[]{R.id.textView1, R.id.textView2, R.id.textView3,
                                                        R.id.textView4});
        listView.setAdapter(adapter);
    } else if (type == R.id.item1) {
```

XML Parser

- SAXFragment.JAVA

```
            textView.setText(parser.parsing1(app.getXml()));
        } else if (type == R.id.item2) {
            textView.setText(parser.parsing2(app.getXml()));
        } else if (type == R.id.item3) {
            textView.setText(parser.parsing3(app.getXml()));
        } else if (type == R.id.item4)
            textView.setText(parser.parsing4(app.getXml()));
        else if (type == R.id.item5)
            textView.setText(parser.parsing5(app.getXml()));
    }
}
```

XML Parser

- MySAXParser.JAVA

```
public class MySAXParser {
    private Context context;

    public MySAXParser(Context context) {
        this.context = context;
    }

    public String parsing1(String xml) {
        SAXParser parser = makeSAX();
        InputStream stream = new ByteArrayInputStream(
                                    xml.getBytes(StandardCharsets.UTF_8));
        SAXHandler1 handler = new SAXHandler1();
        try {
            parser.parse(stream, handler);
        } catch (IOException | SAXException e) {
            Toast.makeText(context, e.getMessage(), Toast.LENGTH_SHORT).show();
        }
        return handler.getResult();
    }
```

XML Parser

- MySAXParser.JAVA

```
public String parsing2(String xml) {
    SAXParser parser = makeSAX();
    InputStream stream = new ByteArrayInputStream(
                                        xml.getBytes(StandardCharsets.UTF_8));
    SAXHandler2 handler = new SAXHandler2();
    try {
        parser.parse(stream, handler);
    } catch (IOException | SAXException e) {
        Toast.makeText(context, e.getMessage(), Toast.LENGTH_SHORT).show();
    }
    return handler.getResult();
}
```

XML Parser

- MySAXParser.JAVA

```
public String parsing3(String xml) {
    SAXParser parser = makeSAX();
    InputStream stream = new ByteArrayInputStream(
                                    xml.getBytes(StandardCharsets.UTF_8));
    SAXHandler3 handler = new SAXHandler3();
    try {
        parser.parse(stream, handler);
    } catch (IOException | SAXException e) {
        Toast.makeText(context, e.getMessage(), Toast.LENGTH_SHORT).show();
    }
    return handler.getResult();
}
```

XML Parser

- MySAXParser.JAVA

```
public String parsing4(String xml) {
    SAXParser parser = makeSAX();
    InputStream stream = new ByteArrayInputStream(
                                    xml.getBytes(StandardCharsets.UTF_8));
    SAXHandler4 handler = new SAXHandler4();
    try {
        parser.parse(stream, handler);
    } catch (IOException | SAXException e) {
        Toast.makeText(context, e.getMessage(), Toast.LENGTH_SHORT).show();
    }
    return handler.getResult();
}
```

XML Parser

- MySAXParser.JAVA

```
public String parsing5(String xml) {
    SAXParser parser = makeSAX();
    InputStream stream = new ByteArrayInputStream(
                                        xml.getBytes(StandardCharsets.UTF_8));
    SAXHandler5 handler = new SAXHandler5();
    try {
        parser.parse(stream, handler);
    } catch (IOException | SAXException e) {
        Toast.makeText(context, e.getMessage(), Toast.LENGTH_SHORT).show();
    }
    return handler.getResult();
}
```

XML Parser

- MySAXParser.JAVA

```
public ArrayList<Country> parsing6(String xml) {
    SAXParser parser = makeSAX();
    InputStream stream = new ByteArrayInputStream(
                                        xml.getBytes(StandardCharsets.UTF_8));
    SAXHandler6 handler = new SAXHandler6();
    try {
        parser.parse(stream, handler);
    } catch (IOException | SAXException e) {
        Toast.makeText(context, e.getMessage(), Toast.LENGTH_SHORT).show();
    }
    return handler.getResult();
}
```

XML Parser

- MySAXParser.JAVA

```
public ArrayList<HashMap<String, String>> parsing7(String xml) {
    SAXParser parser = makeSAX();
    InputStream stream = new ByteArrayInputStream(
                                    xml.getBytes(StandardCharsets.UTF_8));
    SAXHandler7 handler = new SAXHandler7();
    try {
        parser.parse(stream, handler);
    } catch (IOException | SAXException e) {
        Toast.makeText(context, e.getMessage(), Toast.LENGTH_SHORT).show();
    }
    return handler.getResult();
}
```

XML Parser

- MySAXParser.JAVA

```
    private SAXParser makeSAX() {
        SAXParser parser = null;
        try {
            SAXParserFactory factory = SAXParserFactory.newInstance();
            parser = factory.newSAXParser();
        } catch (SAXException | ParserConfigurationException e) {
            Toast.makeText(context, e.getMessage(), Toast.LENGTH_SHORT).show();
        }
        return parser;
    }
}
```

XML Parser

- SAXHandler1.JAVA

```
public class SAXHandler1 extends DefaultHandler {
    private StringBuffer buffer;
    private String team;
    private String text;

    public SAXHandler1() {
        buffer = new StringBuffer();
    }

    public void startElement(String uri, String localName, String qName,
                                                        Attributes atts) {
        if (localName.equals("team"))
            team = " 팀명  = " + atts.getValue(0) + "₩n";
        else if (localName.equals("baseball"))
            buffer.append("프로야구 팀₩n");
    }
```

XML Parser

- SAXHandler1.JAVA

```
    public void characters(char[] chars, int start, int length) {
        text = new String(chars, start, length);
    }

    public void endElement(String uri, String localName, String qName) {
        if (localName.equals("team")) {
            buffer.append(team);
            buffer.append(" 감독 : " + text + "₩n₩n");
        }
    }

    public String getResult() {
        return buffer.toString();
    }
}
```

XML Parser

- SAXHandler2.JAVA

```
public class SAXHandler2 extends DefaultHandler {
    private StringBuffer buffer;
    private String[] att;
    private String text;
    private String name;

    public SAXHandler2() {
        buffer = new StringBuffer();
    }

    @Override
    public void startElement(String uri, String localName, String qName,
                                                        Attributes atts) {
        if (localName.equals("item")) {
            att = new String[atts.getLength()];
            att[0] = " " + atts.getQName(0) + " : " + atts.getValue(0) + '₩n';
            att[1] = " " + atts.getQName(1) + " : " + atts.getValue(1) + " 원₩n";
        }
    }
```

XML Parser

- SAXHandler2.JAVA

```
    @Override
    public void characters(char[] ch, int start, int length) {
            text = new String(ch, start, length);
    }
    @Override
    public void endElement(String uri, String localName, String qName) {
        if (localName.equals("name"))
            name = "품목 : " + text + "₩n";
        if (localName.equals("part") ) {
            buffer.append("-------------------------------₩n");
            buffer.append(name);
            buffer.append(att[0] + att[1]);
            buffer.append("-------------------------------₩n₩n");
        }
    }
    public String getResult() {
        return buffer.toString();
    }
}
```

XML Parser

- SAXHandler3.JAVA

```
public class SAXHandler3 extends DefaultHandler {
    private StringBuilder builder;
    private String num;
    private String text;
    String name;
    String age;
    String depart;

    public SAXHandler3() {
        builder = new StringBuilder();
    }

    @Override
    public void startElement(String uri, String localName, String qName,
                                                    Attributes attributes) {
        if (localName.equals("name"))
            num = attributes.getValue(0);
    }
```

XML Parser

- SAXHandler3.JAVA

```
@Override
public void characters(char[] ch, int start, int length) {
    text = new String(ch, start, length);
}

@Override
public void endElement(String uri, String localName, String qName) {
    Calendar calendar = new GregorianCalendar(Locale.KOREA);
    switch (localName) {
        case "name":
            name = text;
            break;
        case "birth":
            age = calendar.get(Calendar.YEAR) - Integer.parseInt(text) + " 세";
            break;
```

XML Parser

- SAXHandler3.JAVA

```
            case "department":
                depart = text;
                break;
            case "student":
                builder.append(name + "  " + age + "  " + num + "  " + depart + "₩n");
        }
    }

    public String getResult() {
        return builder.toString();
    }
}
```

XML Parser

- SAXHandler4.JAVA

```
public class SAXHandler4 extends DefaultHandler {
    private StringBuilder builder;
    private String text;
    private String name;
    private String email;
    private String title;

    public SAXHandler4() {
        builder = new StringBuilder();
    }

    @Override
    public void startElement(String uri, String localName, String qName,
                                                        Attributes attributes) {
        if (localName.equals("contact"))
            email = attributes.getValue(0);
    }
```

XML Parser

- SAXHandler4.JAVA

```
@Override
public void characters(char[] ch, int start, int length) {
    text = new String(ch, start, length);
}

@Override
public void endElement(String uri, String localName, String qName) {
    switch (localName) {
        case "name":
            name = text;
            break;
        case "title":
            title = text;
            break;
        case "member":
            builder.append("이름 : " + name + "₩n 직위 : " + title +
                    "₩n E-mail : " + email + "₩n₩n");
    }
}
```

XML Parser

- SAXHandler4.JAVA

```
    public String getResult() {
        return builder.toString();
    }
}
```

XML Parser

- SAXHandler5.JAVA

```
public class SAXHandler5 extends DefaultHandler {
    private StringBuffer buffer;
    private String[] sedan;
    private String text;
    private String name;

    public SAXHandler5() {
        buffer = new StringBuffer();
    }

    public void startElement(String uri, String localName, String qName,
                                                        Attributes atts) {
        if (localName.equals("sedan")) {
            sedan = new String[atts.getLength()];
            sedan[0] = " " + atts.getQName(0) + " = " + atts.getValue(0) + "₩n";
            sedan[1] = " " + atts.getQName(1) + " = " + atts.getValue(1) + "₩n";
            sedan[2] = " " + atts.getQName(2) + " = " + atts.getValue(2) + " 만원₩n";
        }
    }
```

XML Parser

- SAXHandler5.JAVA

```
    public void characters(char[] chars, int start, int length) {
        text = new String(chars, start, length);
    }

    public void endElement(String uri, String localName, String qName) {
        if (localName.equals("product"))
            name = text + '₩n';
        else if (localName.equals("items")) {
            buffer.append("Model Name : " + name + sedan[0] + sedan[1]
                                                    +  sedan[2] + "₩n");

        }
    }

    public String getResult() {
        return buffer.toString();
    }
}
```

XML Parser

- SAXHandler6.JAVA

```
public class SAXHandler6 extends DefaultHandler {
    private ArrayList<Country> countries;
    private String text;
    private Country country;

    public SAXHandler6() {
        countries = new ArrayList<>();
    }

    public void startElement(String uri, String localName, String qName,
                                                        Attributes attributes) {
        if (localName.equals("nation"))
            country = new Country();
    }

    @Override
    public void characters(char[] ch, int start, int length) {
        text = new String(ch, start, length);
    }
```

XML Parser

- SAXHandler6.JAVA

```
@Override
public void endElement(String uri, String localName, String qName) {
        if (localName.equals("name"))
            country.setCountry(text);
        else if (localName.equals("flag"))
            country.setFlag(text);
        else if (localName.equals("lang"))
            country.setLang(text);
        else if (localName.equals("capital"))
            country.setCapital(text);
        else if (localName.equals("code"))
            country.setCode(text);
        else if (localName.equals("currencyname"))
            country.setCurrency(text);
        else if (localName.equals("nation"))
            countries.add(country);
}
```

XML Parser

- SAXHandler6.JAVA

```
    public ArrayList<Country> getResult() {
        return countries;
    }
}
```

XML Parser

- SAXHandler7.JAVA

```
public class SAXHandler7 extends DefaultHandler {
    private ArrayList<HashMap<String, String>> userList;
    private String text;
    private HashMap<String, String> user;

    public SAXHandler7() {
        userList = new ArrayList<>();
    }

    @Override
    public void startElement(String uri, String localName, String qName,
                                                        Attributes attributes) {
        if (localName.equals("PART"))
            user = new HashMap<>();
    }
    @Override
    public void characters(char[] ch, int start, int length) {
        text = new String(ch, start, length);
    }
```

XML Parser

- SAXHandler7.JAVA

```
@Override
public void endElement(String uri, String localName, String qName) {
    switch (localName) {
        case "ITEM":
            user.put("ITEM", "아이템 : " + text);
            break;
        case "MANUFACTURER":
            user.put("MANUFACTURER", "제조사 : " + text);
            break;
        case "MODEL":
            user.put("MODEL", "모델 : " + text);
            break;
        case "COST":
            user.put("COST", "가격 : " + text);
            break;
        case "PART":
            userList.add(user);
    }
}
```

XML Parser

- SAXHandler7.JAVA

```
    public ArrayList<HashMap<String, String>> getResult() {
        return userList;
    }
}
```

XML Parsing

- fragment_parse1.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <TextView
        android:id="@+id/textView"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="#CAD79A"
        android:paddingTop="10dp"
        android:text="파싱 결과가 여기에 표시됩니다."
        android:textSize="20dp" />
</LinearLayout>
```

XML Parsing

- fragment_parse2.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <ListView
        android:id="@+id/listView"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="#EBD047" />
</LinearLayout>
```

XML Parser

- PULLFragment.JAVA

```
public class PULLFragment extends Fragment {
    private MyApplication app;
    private Activity activity;
    private int type;
    private ListView listView;
    private TextView textView;

    public PULLFragment(Activity activity, int type) {
        this.activity = activity;
        this.type = type;
        app = (MyApplication) activity.getApplication();
    }
```

XML Parser

- PULLFragment.JAVA

```
@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
                                                    Bundle savedInstanceState) {
    ViewGroup rootView;
    if (type == R.id.item6 || type == R.id.item7) {
        rootView = (ViewGroup) inflater.inflate(R.layout.fragment_parse2,
                                                        container, false);
        listView = rootView.findViewById(R.id.listView);
    } else {
        rootView = (ViewGroup) inflater.inflate(R.layout.fragment_parse1,
                                                            container, false);
        textView = rootView.findViewById(R.id.textView);
    }

    return rootView;
}
```

XML Parser

- PULLFragment.JAVA

```
@Override
public void onViewCreated(@NonNull View view,
                                        @Nullable Bundle savedInstanceState) {
    PULLParser parser = new PULLParser(activity);
    if (type == R.id.item6) {
        ArrayList<Country> countries = parser.parsing7(app.getXml());
        ArrayAdapter<Country> adapter = new CountryAdapter(activity,
                R.layout.country, countries);
        listView.setAdapter(adapter);
    } else if (type == R.id.item7) {
        ArrayList<HashMap<String, String>> items = parser.parsing8(app.getXml());
        SimpleAdapter adapter = new SimpleAdapter(activity, items,
                R.layout.list_item,
                new String[]{"ITEM", "제조사", "MODEL", "COST"},
                new int[]{R.id.textView1, R.id.textView2, R.id.textView3,
                                                        R.id.textView4});
        listView.setAdapter(adapter);
    } else if (type == R.id.item1) {
```

XML Parser

- PULLFragment.JAVA

```
            textView.setText(parser.parsing1(app.getXml()));
        } else if (type == R.id.item2) {
            textView.setText(parser.parsing2(app.getXml()));
        } else if (type == R.id.item3) {
            textView.setText(parser.parsing3(app.getXml()));
        } else if (type == R.id.item4)
            textView.setText(parser.parsing5(app.getXml()));
        else if (type == R.id.item5)
            textView.setText(parser.parsing6(app.getXml()));
    }
}
```

XML Parser

- PULLParser.JAVA

```
public class PULLParser {
    private Context context;

    public PULLParser(Context context) {
        this.context = context;
    }

    public String parsing1(String xml) {
        StringBuilder builder = new StringBuilder();
        XmlPullParser parser = makePULL(xml);
        String text = null;
        String team = null;
        try {
            int eventType = parser.getEventType();
            while (eventType != XmlPullParser.END_DOCUMENT) {
                String tag = parser.getName();
```

XML Parser

- PULLParser.JAVA

```
switch (eventType) {
    case XmlPullParser.START_TAG:
        if (tag.equals("team"))
            team = " 팀명 = " + parser.getAttributeValue(0) + "₩n";
        else if (tag.equals("baseball"))
            builder.append("프로야구 팀₩n");
        break;
    case XmlPullParser.TEXT:
        text = parser.getText();
        break;
    case XmlPullParser.END_TAG:
        if (tag.equals("team")) {
            builder.append(team);
            builder.append(" 감독 : " + text + "₩n₩n");
        }
    }
    eventType = parser.next();
}
```

XML Parser

- PULLParser.JAVA

```
    } catch (XmlPullParserException | IOException e) {
        Toast.makeText(context, e.getMessage(), Toast.LENGTH_SHORT).show();
    }
    return builder.toString();
}

public String parsing2(String xml) {
    StringBuilder builder = new StringBuilder();
    XmlPullParser parser = makePULL(xml);
    String text = "";
    String name = "";
    String[] attr = null;
    try {
        int eventType = parser.getEventType();
        while (eventType != XmlPullParser.END_DOCUMENT) {
            String tag = parser.getName();
```

XML Parser

- PULLParser.JAVA

```
switch (eventType) {
    case XmlPullParser.START_TAG:
        if (tag.equals("item")) {
            attr = new String[parser.getAttributeCount()];
            attr[0] = " " + parser.getAttributeName(0) + " : "
                                    + parser.getAttributeValue(0) + "₩n";
            attr[1] = " " + parser.getAttributeName(1) + " : "
                                    + parser.getAttributeValue(1) + " 원₩n";
        }
        break;
    case XmlPullParser.TEXT:
        text = parser.getText() + "₩n";
        break;
    case XmlPullParser.END_TAG:
        if (tag.equals("name"))
            name = "품목 : " + text;
        else if (tag.equals("part")) {
```

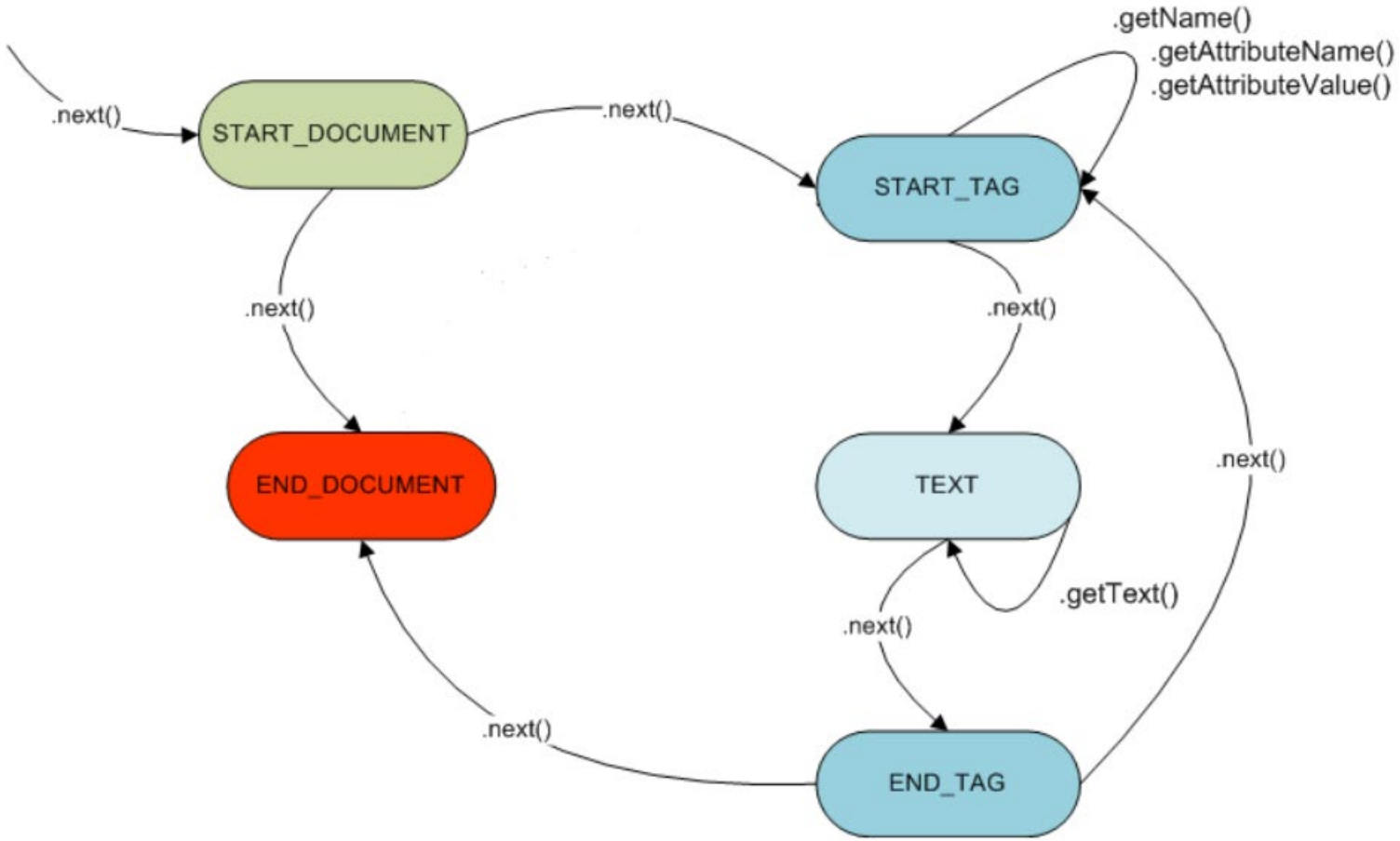
XML Parser

- PULLParser.JAVA

```
                    builder.append("----------------------------------₩n");
                    builder.append(name);
                    builder.append(attr[0] + attr[1]);
                    builder.append("----------------------------------₩n₩n");
                }
            }
            eventType = parser.next();
        }
    } catch (XmlPullParserException | IOException e) {
        Toast.makeText(context, e.getMessage(), Toast.LENGTH_SHORT).show();
    }
    return builder.toString();
}
```

XML PULL Parsing

- XML PULL Parser Lifecycle



XML PULL Parsing

- XMLPullParser는 문서를 순회하면서 next() 메소드를 사용하여 다음과 같은 주요 eventType을 감지
  - START_TAG
  - TEXT
  - END_TAG
  - END_DOCUMENT
- Tag의 시작 부분이 인식되면, getName() 메소드를 사용하여 Tag 이름을 읽어 옴
- TEXT Event 이후 Data를 추출하기 위해서는 getText() 메소드를 사용하여 Data를 문자열로 읽어 옴

XML PULL Parsing

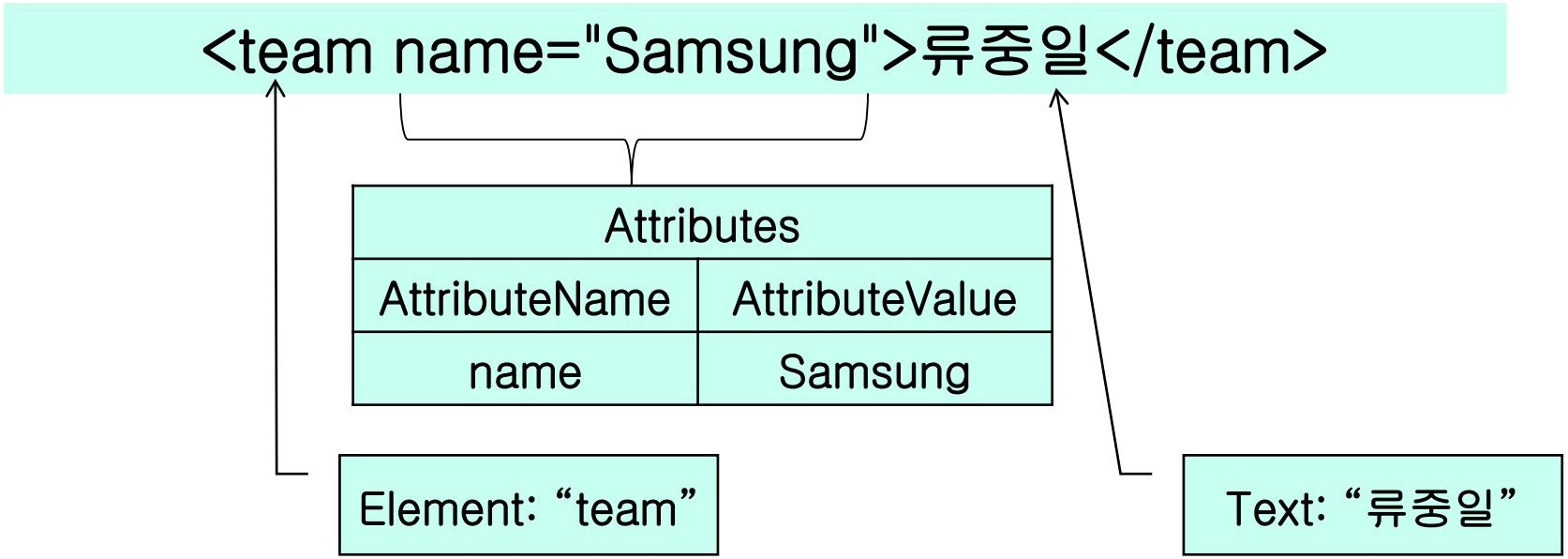
- XmlPullParser는 문서를 순차적으로 읽으면서 Event를 발생시키므로 뒤로 가지는 못함
- 따라서 XmlPullParser.START_TAG 발생시 임시 변수를 선언하여 Tag값을 저장
- TEXT Event 시 임시 변수에 저장된 Tag값을 확인하여 XmlPullParser.TEXT 이벤트가 발생한 위치(Tag)가 어디인지를 구분하고 적절하게 대처
- XmlPullParser.END_TAG에서는 임시 변수를 초기화(null) 함
- XmlPullParser.START_TAG와 XmlPullParser.END_TAG Event에서는 getName() 메소드를 사용. getText()사용시 null 반환
- XmlPullParser.TEXT Event시에는 getText() 메소드를 사용, getName()사용시 null 반환

XML PULL Parsing

- XmlPullParser.TEXT Event는 Text가 존재하지 않아도 발생
  - 예) <data></data>
- Tag 내에 존재하는 속성값은 XmlPullParser.START_TAG Event시 추출
  - 예) <data size=100>test</data>

XML PULL Parsing

- <element>의 내부 속성은 다음 메소드를 사용하여 추출 할 수 있음
  - getAttributeCount()
  - getAttributeName()
  - getAttributeValue()



XML Parser

- PULLParser.JAVA

```
public String parsing3(String xml) {
    StringBuilder builder = new StringBuilder();
    XmlPullParser parser = makePULL(xml);
    String text = null;
    String num = null;
    String name = null;
    String age = null;
    String depart = null;
    Calendar calendar = new GregorianCalendar(Locale.KOREA);
    try {
        int eventType = parser.getEventType();
        while (eventType != XmlPullParser.END_DOCUMENT) {
            String tag = parser.getName();
            switch (eventType) {
                case XmlPullParser.START_TAG:
                    if (tag.equals("name"))
                        num = parser.getAttributeValue(0);
                    break;
```

XML Parser

- PULLParser.JAVA

```
case XmlPullParser.TEXT:
    text = parser.getText();
    break;
case XmlPullParser.END_TAG:
    switch (tag) {
        case "name":
            name = text;
            break;
        case "birth":
            age = calendar.get(Calendar.YEAR) - Integer.parseInt(text) + " 세"
            break;
        case "department":
            depart = text;
            break;
        case "student":
            builder.append(name + "  " + age + "  " + num + "  " +
                    depart + "₩n");
    }
}
```

XML Parser

- PULLParser.JAVA

```
            eventType = parser.next();
        }
    } catch (XmlPullParserException | IOException e) {
        Toast.makeText(context, e.getMessage(), Toast.LENGTH_SHORT).show();
    }

    return builder.toString();
}

public String parsing4(Resources resources, int id) {
    StringBuilder builder = new StringBuilder();
    XmlResourceParser parser = resources.getXml(id);
    String text = "";
    String color = "";
    try {
        int eventType = parser.getEventType();
        while (eventType != XmlPullParser.END_DOCUMENT) {
            String tag = parser.getName();
```

XML Parser

- PULLParser.JAVA

```
            switch (eventType) {
                case XmlPullParser.START_TAG:
                    if (tag.equals("cat"))
                        color = parser.getAttributeValue(0);
                    break;
                case XmlPullParser.TEXT:
                    text = parser.getText();
                    break;
                case XmlPullParser.END_TAG:
                    if (tag.equals("cat"))
                        builder.append(String.format("고양이 : %s (%s)₩n", text, color));
            }
            eventType = parser.next();
        }
    } catch (XmlPullParserException | IOException e) {
        Toast.makeText(context, e.getMessage(), Toast.LENGTH_SHORT).show();
    }
    return builder.toString();
}
```

XML Parser

- PULLParser.JAVA

```
public String parsing5(String xml) {
    StringBuilder builder = new StringBuilder();
    XmlPullParser parser = makePULL(xml);
    String text = "";
    String name = "";
    String email = "";
    String title = "";
    try {
        int eventType = parser.getEventType();
        while (eventType != XmlPullParser.END_DOCUMENT) {
            String tag = parser.getName();
            switch (eventType) {
                case XmlPullParser.START_TAG:
                    if (tag.equals("contact"))
                        email = parser.getAttributeValue(0);
                    break;
                case XmlPullParser.TEXT:
                    text = parser.getText();
                    break;
```

XML Parser

```
                case XmlPullParser.END_TAG:
                    switch (tag) {
                        case "name":
                            name = text;
                            break;
                        case "title":
                            title = text;
                            break;
                        case "member":
                            builder.append("이름 : " + name + "₩n 직위 : "
                                    + title + "₩n E-mail : " + email + "₩n₩n");
                    }
                }
                eventType = parser.next();
            }
        } catch (IOException | XmlPullParserException e) {
            Toast.makeText(context, e.getMessage(), Toast.LENGTH_SHORT).show();
        }
        return builder.toString();
    }
```

XML Parser

- PULLParser.JAVA

```
public String parsing6(String xml) {
    StringBuilder builder = new StringBuilder();
    XmlPullParser parser = makePULL(xml);
    String text = "";
    String name = "";
    String type = "";
    String cc = "";
    String price = "";
    try {
        int eventType = parser.getEventType();
        while (eventType != XmlPullParser.END_DOCUMENT) {
            String tag = parser.getName();
            if (eventType == XmlPullParser.START_TAG) {
                if (tag.equals("sedan")) {
                    type = parser.getAttributeValue(0);
                    cc = parser.getAttributeValue(1);
                    price = parser.getAttributeValue(2);
                }
```

XML Parser

- PULLParser.JAVA

```
            } else if (eventType == XmlPullParser.TEXT)
                text = parser.getText();
            else if (eventType == XmlPullParser.END_TAG) {
                if (tag.equals("product"))
                    name = text;
                else if (tag.equals("items"))
                    builder.append("이름 : " + name + "₩nType : " + type
                            + "₩nCC : " + cc + "₩nPrice : " + price + "₩n₩n");
            }
            eventType = parser.next();
        }
    } catch (IOException | XmlPullParserException e) {
        Toast.makeText(context, e.getMessage(), Toast.LENGTH_SHORT).show();
    }
    return builder.toString();
}
```

XML Parser

- PULLParser.JAVA

```
public ArrayList<Country> parsing7(String xml) {
    ArrayList<Country> countries = new ArrayList<>();
    XmlPullParser parser = makePULL(xml);
    String text = "";
    Country country = null;
    try {
        int event = parser.getEventType();
        while (event != XmlPullParser.END_DOCUMENT) {
            String tag = parser.getName();
            switch (event) {
                case XmlPullParser.START_TAG:
                    if (tag.equals("nation"))
                        country = new Country();
                    break;
                case XmlPullParser.TEXT:
                    text = parser.getText();
                    break;
```

XML Parser

- PULLParser.JAVA

```
            case XmlPullParser.END_TAG:
                if (tag.equals("name"))
                    country.setCountry(text);
                else if (tag.equals("flag"))
                    country.setFlag(text);
                else if (tag.equals("lang"))
                    country.setLang(text);
                else if (tag.equals("capital"))
                    country.setCapital(text);
                else if (tag.equals("code"))
                    country.setCode(text);
                else if (tag.equals("currencyname"))
                    country.setCurrency(text);
                else if (tag.equals("nation"))
                    countries.add(country);
            }
            event = parser.next();
        }
```

XML Parser

- PULLParser.JAVA

```
    } catch (IOException | XmlPullParserException e) {
        Toast.makeText(context, e.getMessage(), Toast.LENGTH_SHORT).show();
    }
    return countries;
}

public ArrayList<HashMap<String, String>> parsing8(String xml) {
    ArrayList<HashMap<String, String>> itemList = new ArrayList<>();
    XmlPullParser parser = makePULL(xml);
    HashMap<String, String> item = null;
    String text = "";
    try {
        int eventType = parser.getEventType();
        while (eventType != XmlPullParser.END_DOCUMENT) {
            String tag = parser.getName();
            switch (eventType) {
                case XmlPullParser.START_TAG:
                    if (tag.equals("PART"))
                        item = new HashMap<>();
```

XML Parser

- PULLParser.JAVA

```
case XmlPullParser.TEXT:
    text = parser.getText();
    break;
case XmlPullParser.END_TAG:
    switch (tag) {
        case "ITEM":
            item.put("ITEM", "아이템 : " + text);
            break;
        case "MANUFACTURER":
            item.put("MANUFACTURER", "제조사 : " + text);
            break;
        case "MODEL":
            item.put("MODEL", "모델 : " + text);
            break;
        case "COST":
            item.put("COST", "가격 : " + text);
            break;
```

XML Parser

- PULLParser.JAVA

```
                    case "PART":
                        itemList.add(item);
                }
            }
            eventType = parser.next();
        }
    } catch (XmlPullParserException | IOException e) {
        Toast.makeText(context, e.getMessage(), Toast.LENGTH_SHORT).show();
    }
    return itemList;
}
```

XML Parser

- PULLParser.JAVA

```
    private XmlPullParser makePULL(String xml) {
        InputStream stream = new ByteArrayInputStream(
                                                xml.getBytes(StandardCharsets.UTF_8));
        XmlPullParser parser;
        try {
            XmlPullParserFactory factory = XmlPullParserFactory.newInstance();
            parser = factory.newPullParser();
            parser.setInput(stream, "UTF-8");
        } catch (XmlPullParserException e) {
            throw new RuntimeException(e);
        }
        return parser;
    }
}
```

XML Parsing

- Country.JAVA

```
public class Country {
    private String country;
    private String flag;
    private String lang;
    private String capital;
    private String currency;
    private String code;

    public String getCode() {
        return code;
    }

    public String getFlag() {
        return flag;
    }

    public void setFlag(String flag) {
        this.flag = flag;
    }
```

XML Parsing

- Country.JAVA

```
public void setCode(String code) {
    this.code = code;
}

public String getLang() {
    return lang;
}

public void setLang(String lang) {
    this.lang = lang;
}

public String getCapital() {
    return capital;
}

public void setCapital(String capital) {
    this.capital = capital;
}
```

XML Parsing

- Country.JAVA

```
    public String getCurrency() {
        return currency;
    }

    public void setCurrency(String currency) {
        this.currency = currency;
    }

    public String getCountry() {
        return country;
    }

    public void setCountry(String country) {
        this.country = country;
    }
}
```

XML Parsing

- CountryAdapter.JAVA

```
public class CountryAdapter extends ArrayAdapter<Country> {
    private Context context;
    public CountryAdapter(@NonNull Context context, int resource,
                                                    List<Country> countries) {
        super(context, resource, countries);
        this.context = context;
    }

    @NonNull
    @Override
    public View getView(int position, @Nullable View convertView,
                                                @NonNull ViewGroup parent) {
        ViewHolder holder;
        LayoutInflater inflater = (LayoutInflater) context.getSystemService(
                                                Activity.LAYOUT_INFLATER_SERVICE);
        if (convertView == null) {
            convertView = inflater.inflate(R.layout.country, parent, false);
            holder = new ViewHolder(convertView);
            convertView.setTag(holder);
```

XML Parsing

- CountryAdapter.JAVA

```
    } else
        holder = (ViewHolder) convertView.getTag();
    Country country = getItem(position);
    holder.textView1.setText(" 국가 : " + country.getCountry());
    holder.textView2.setText(" 수도 : " + country.getCapital()
                                    + "사용 언어 : " + country.getLang());
    holder.textView3.setText(" 화폐 : " + country.getCurrency() + " ("
                                            + country.getCode() +")");
    Glide.with(context).load(country.getFlag())
                .placeholder(R.drawable.ic_launcher).into(holder.imageView);

    return convertView;
}
```

XML Parsing

- CountryAdapter.JAVA

```
    private class ViewHolder {
        private TextView textView1;
        private TextView textView2;
        private TextView textView3;
        private ImageView imageView;

        public ViewHolder(View v) {
            textView1 = v.findViewById(R.id.textView1);
            textView2 = v.findViewById(R.id.textView2);
            textView3 = v.findViewById(R.id.textView3);
            imageView = v.findViewById(R.id.imageView);
        }
    }
}
```

XML Parsing

- country.xml

```
<?xml version="1.0" encoding="utf-8" ?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="5dp">

    <LinearLayout
        android:layout_width="0dp"
        android:layout_height="70dp"
        android:layout_marginLeft="5dp"
        android:layout_weight="1"
        android:gravity="center_vertical"
        android:orientation="vertical">

        <TextView
            android:id="@+id/textView1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Country : " />
```

XML Parsing

- country.xml

```
    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="언어 : " />

    <TextView
        android:id="@+id/textView3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Population : " />
</LinearLayout>
```

XML Parsing

- country.xml

```
    <ImageView
        android:id="@+id/imageView"
        android:layout_width="90dp"
        android:layout_height="70dp"
        android:layout_marginLeft="10dp"
        android:layout_marginRight="10dp"
        android:background="@drawable/square_border_black"
        android:padding="2dp"
        android:scaleType="centerCrop" />
</LinearLayout>
```

XML Parsing

- list_item.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:padding="5dp"
    android:orientation="vertical">

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:textColor="#FF0000" />

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceMedium" />
```

XML Parsing

- list_item.xml

```
    <TextView
        android:id="@+id/textView3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceMedium" />

    <TextView
        android:id="@+id/textView4"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceMedium" />
</LinearLayout>
```