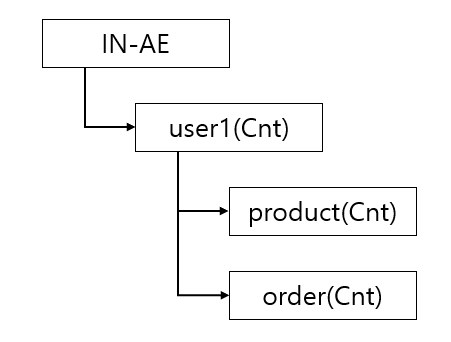
구현

스마트폰 어플리케이션(IN-AE)의 리소스 구조



[그림] 스마트폰 어플리케이션(IN-AE)의 리소스 구조

- 스마트폰 어플리케이션의 리소스 구조는 다음과 같다.

- user1(Cnt) : 유저의 정보를 담고 있는 컨테이너로 하위 컨테이너를 담고 있다.

- product(Cnt) : 주문 목록에 대한 정보를 담고 있는 컨테이너로 하위 contentIntstnace를 보유하고 있다.

- 주문 물품의 이름과 수량, 금액에 대한 정보가 들어있다.

- order(Cnt) : 주문 완료 정보를 담고 있는 컨테이너로 하위 contentInstance를 보유하고 있다.

- ‘0’, ‘1’, ‘2’로 주문 정보를 표현한다.

- ‘0’ : 주문이 완료되지 않은 상태이다.

- ‘1’ : 주문 정보가 들어와서 서버에서 처리하고 있는 상태이다.

- ‘2’ : 주문이 완료된 상태이다.

엔티티의 역할

스마트폰 어플리케이션(IN-AE)

- 스마트폰 어플리케이션은 IN-AE로 모델링 되며, oneM2M 서비스 플랫폼인 IN-CSE와 직접 통신할 수 있고 다음과 같은 과정을 통해 모델링 된다.

스마트폰 주문 어플리케이션 초기화

IN-CSE에 스마트폰 어플리케이션 등록

어플로부터 주문을 받아 상품 품목, 주문 컨테이너에 cin 생성을 실행

IN-CSE에 주문 정보를 주기적으로 요청하여 주문 완료 여부를 확인.

주문이 완료되면 상태 표시줄에 알림을 띄우고 진동이 울림

구현 절차

어플리케이션 엔티티 등록

스마트폰 어플리케이션(IN-AE)

HTTP Request:  
 POST /Mobius HTTP/1.1  
 Host: 143.85.132.36:7579  
 X-M2M-Origin: CAE  
 Content-Type: application/json; ty=2  
 Accept: application/json  
 X-M2M-RI: 1234  
 {  
 "m2m:ae" : {  
 "rn": app1,  
 "api":application,  
 "rr": true  
 }  
 }

어플리케이션 코드

private void requestAECreation(){  
 JSONObject jObj = new JSONObject();  
 JSONObject tmp = new JSONObject();  
  
 try{  
 tmp.put("rn", "app1");  
 tmp.put("api", "application");  
 tmp.put("rr", true);  
 jObj.put("m2m:ae", tmp);  
 } catch(JSONException e1){  
 e1.printStackTrace();  
 }  
  
 try{  
 String host\_url = "http://146.56.166.36:7579/Mobius";  
 HttpURLConnection conn = null;  
  
 URL url = new URL(host\_url);  
 conn = (HttpURLConnection)url.openConnection();  
  
 conn.setRequestMethod("POST");  
 conn.setRequestProperty("Content-type", "application/json; ty=2");  
 conn.setRequestProperty("Accept", "application/json");  
 conn.setRequestProperty("X-M2M-RI", "1234");  
 conn.setRequestProperty("X-M2M-Origin", "CAE1");  
  
 conn.setDoOutput(true);  
 conn.setDoInput(true);  
 BufferedWriter bw = new BufferedWriter(new OutputStreamWriter(conn.getOutputStream()));  
  
 bw.write(jObj.toString());  
 bw.flush();  
 bw.close();  
  
 int responseCode = conn.getResponseCode();  
 }catch(IOException ie){  
 ie.printStackTrace();  
 }catch(Exception ee){  
 ee.printStackTrace();  
 }  
}

컨테이너 생성

IN-AE의 컨테이너 생성

HTTP Request:  
 POST /Mobius/app1 HTTP/1.1  
 Host: 143.85.132.35:7579  
 X-M2M-Origin: CAE  
 Content-Type: application/json; ty=3  
 Accept: application/json  
 X-M2M-RI: 1234  
 {  
 “m2m:cnt” : {  
 “rn” : “user1”,  
 “mni” : 100  
 }  
 }

어플리케이션 코드

JSONObject jObj = new JSONObject();  
JSONObject tmp = new JSONObject();  
try{  
 tmp.put("rn", "user1");  
 tmp.put("mni", 100);  
 jObj.put("m2m:cnt", tmp);  
} catch(JSONException e1){  
 e1.printStackTrace();  
}  
  
try{  
 String host\_url = "http://146.56.166.36:7579/Mobius/app1";  
 HttpURLConnection conn = null;  
  
 URL url = new URL(host\_url);  
 conn = (HttpURLConnection)url.openConnection();  
  
 conn.setRequestMethod("POST");  
 conn.setRequestProperty("Content-type", "application/json; ty=3");  
 conn.setRequestProperty("Accept", "application/json");  
 conn.setRequestProperty("X-M2M-RI", "1234");  
 conn.setRequestProperty("X-M2M-Origin", "CAE1");  
  
 conn.setDoOutput(true);  
 BufferedWriter bw = new BufferedWriter(new OutputStreamWriter(conn.getOutputStream()));  
  
 bw.write(jObj.toString());  
 bw.flush();  
 bw.close();  
  
 int responseCode = conn.getResponseCode();  
}catch(IOException ie){  
 ie.printStackTrace();  
}catch(Exception ee){  
 ee.printStackTrace();  
}

사용자 품목 하위에 주문 품목 관련 컨테이너와 주문 정보 관련 컨테이너를  
생성한다.

HTTP Request:  
 POST /Mobius/app1/user1 HTTP/1.1  
 Host: 143.85.132.35:7579  
 X-M2M-Origin: CAE  
 Content-Type: application/json; ty=3  
 X-M2M-RI: 1234  
 {  
 “m2m:cnt” : {  
 “rn” : “product”,  
 “mni” : 100  
 }  
 }

HTTP Request:  
 POST /Mobius/app1/user1 HTTP/1.1  
 Host: 143.85.132.35:7579  
 X-M2M-Origin: CAE  
 Content-Type: application/json; ty=3  
 X-M2M-RI: 1234  
 {  
 “m2m:cnt” : {  
 “rn” : “order”,  
 “mni” : 100  
 }  
 }

어플리케이션 코드

품목 관련 컨테이너 생성 코드

jObj = new JSONObject();  
tmp = new JSONObject();  
try{  
 tmp.put("rn", "product");  
 tmp.put("mni", 100);  
 jObj.put("m2m:cnt", tmp);  
} catch(JSONException e1){  
 e1.printStackTrace();  
}  
  
try{  
 String host\_url = "http://146.56.166.36:7579/Mobius/app1/user1";  
 HttpURLConnection conn = null;  
  
 URL url = new URL(host\_url);  
 conn = (HttpURLConnection)url.openConnection();  
  
 conn.setRequestMethod("POST");  
 conn.setRequestProperty("Content-type", "application/json; ty=3");  
 conn.setRequestProperty("Accept", "application/json");  
 conn.setRequestProperty("X-M2M-RI", "1234");  
 conn.setRequestProperty("X-M2M-Origin", "CAE1");  
  
 conn.setDoOutput(true);  
 BufferedWriter bw = new BufferedWriter(new OutputStreamWriter(conn.getOutputStream()));  
  
 bw.write(jObj.toString());  
 bw.flush();  
 bw.close();  
  
 int responseCode = conn.getResponseCode();  
}catch(IOException ie){  
 ie.printStackTrace();  
}catch(Exception ee){  
 ee.printStackTrace();  
}

주문 정보 관련 컨테이너 생성 코드

jObj = new JSONObject();  
tmp = new JSONObject();  
try{  
 tmp.put("rn", "order");  
 tmp.put("mni", 100);  
 jObj.put("m2m:cnt", tmp);  
} catch(JSONException e1){  
 e1.printStackTrace();  
}  
  
try{  
 String host\_url = "http://146.56.166.36:7579/Mobius/app1/user1";  
 HttpURLConnection conn = null;  
  
 URL url = new URL(host\_url);  
 conn = (HttpURLConnection)url.openConnection();  
  
 conn.setRequestMethod("POST");  
 conn.setRequestProperty("Content-type", "application/json; ty=3");  
 conn.setRequestProperty("Accept", "application/json");  
 conn.setRequestProperty("X-M2M-RI", "1234");  
 conn.setRequestProperty("X-M2M-Origin", "CAE1");  
  
 conn.setDoOutput(true);  
 BufferedWriter bw = new BufferedWriter(new OutputStreamWriter(conn.getOutputStream()));  
  
 bw.write(jObj.toString());  
 bw.flush();  
 bw.close();  
  
 int responseCode = conn.getResponseCode();  
}catch(IOException ie){  
 ie.printStackTrace();  
}catch(Exception ee){  
 ee.printStackTrace();  
}

ContentInstance 생성

IN-AE의 contentInstance 생성

주문과 함께 주문, 품목 관련 내용을 담은 IN-AE의 컨테이너 아래에 contentInstance 리소스를 생성한다.

1) 품목 관련 contentInstance 리소스 생성 절차

HTTP Request:  
 POST /Mobius/app1/user1/product HTTP/1.1  
 Host: 143.85.132.35:7579  
 X-M2M-Origin: CAE  
 Content-Type: application/json; ty=4  
 Accept: application/json

X-M2M-RI: 1234  
 {  
 “m2m:cin” : {  
 “con” : {  
 “product1” : “품목1”,  
 “quantity1” : (품목 1의 양),  
 “price1” : (품목 1의 금액),  
 “product2” : “품목2”,  
 “quantity2” : (품목 2의 양),  
 “price2” : (품목 2의 금액)  
 }  
 }  
 }

어플리케이션 코드

JSONObject jObj = new JSONObject();  
JSONObject tmp = new JSONObject();  
JSONObject jObj2 = new JSONObject();  
try{  
 tmp.put("product", product);  
 tmp.put("quantity", mQuantity);  
 tmp.put("price", PRICE\_CARROT);  
 tmp.put("product2", product2);  
 tmp.put("quantity2", mQuantity2);  
 tmp.put("price2", PRICE\_COCACOLA);  
 jObj2.put("con", tmp);  
 jObj.put("m2m:cin", jObj2);  
} catch(JSONException e1){  
 e1.printStackTrace();  
}  
  
try{  
 String host\_url = "http://146.56.166.36:7579/Mobius/app1/user1/product";  
 HttpURLConnection conn = null;  
  
 URL url = new URL(host\_url);  
 conn = (HttpURLConnection)url.openConnection();  
  
 conn.setRequestMethod("POST");  
 conn.setRequestProperty("Content-type", "application/json; ty=4");  
 conn.setRequestProperty("X-M2M-Origin", "CAE1");  
 conn.setRequestProperty("X-M2M-RI", "1234");  
 conn.setRequestProperty("Accept", "application/json");  
  
 conn.setDoOutput(true);  
 BufferedWriter bw = new BufferedWriter(new OutputStreamWriter(conn.getOutputStream()));  
  
 bw.write(jObj.toString());  
 bw.flush();  
 bw.close();  
  
 int responseCode = conn.getResponseCode();  
}catch(IOException ie){  
 ie.printStackTrace();  
}catch(Exception ee){  
 ee.printStackTrace();  
}

2) 주문 관련 contentInstance 리소스 생성 절차

HTTP Request:  
 POST /Mobius/app1/user1/order HTTP/1.1  
 Host: 143.85.132.35:7579  
 X-M2M-Origin: CAE  
 Content-Type: application/json; ty=4  
 Accept: application/json  
 X-M2M-RI: 1234  
 {  
 “m2m:cin” : {  
 “con” : {  
 “order” : 0  
 }  
 }  
 }

어플리케이션 코드

jObj = new JSONObject();  
tmp = new JSONObject();  
jObj2 = new JSONObject();  
try{  
 tmp.put("order", 0);  
 jObj2.put("con", tmp);  
 jObj.put("m2m:cin", jObj2);  
} catch(JSONException e1){  
 e1.printStackTrace();  
}  
  
try{  
 String host\_url = "http://146.56.166.36:7579/Mobius/app1/user1/order";  
 HttpURLConnection conn = null;  
  
 URL url = new URL(host\_url);  
 conn = (HttpURLConnection)url.openConnection();  
  
 conn.setRequestMethod("POST");  
 conn.setRequestProperty("Content-type", "application/json; ty=4");  
 conn.setRequestProperty("X-M2M-Origin", "CAE1");  
 conn.setRequestProperty("X-M2M-RI", "1234");  
 conn.setRequestProperty("Accept", "application/json");  
  
 conn.setDoOutput(true);  
 BufferedWriter bw = new BufferedWriter(new OutputStreamWriter(conn.getOutputStream()));  
  
 bw.write(jObj.toString());  
 bw.flush();  
 bw.close();  
  
 int responseCode = conn.getResponseCode();  
}catch(IOException ie){  
 ie.printStackTrace();  
}catch(Exception ee){  
 ee.printStackTrace();  
}

최신 contentInstance 검색

사용자는 가상 리소스 latest (줄여서 la)가 부착된 최신 contentInstance RETRIEVE 요청 보내기를 통해 현재 주문 현황을 검색할 수 있다.

1) IN-AE의 최신 contentInstance 검색

HTTP Request:  
 GET /Mobius/app1/user1/order/la HTTP 1.1  
 Host: 143.85.132.35:7579  
 X-M2M-Origin: CAE1  
 X-M2M-RI: 1234  
 Accept: application/json

어플리케이션 코드

public void getRequest(){  
 Timer timer = new Timer();  
 TimerTask task = new TimerTask(){  
 @Override  
 public void run(){  
 new Thread(){  
 public void run(){  
 try{  
 String host\_url = "http://146.56.166.36:7579/Mobius/app1/user1/order/la";  
 HttpURLConnection conn;  
  
 URL url = new URL(host\_url);  
 conn = (HttpURLConnection)url.openConnection();  
  
 conn.setRequestMethod("GET");  
 conn.setRequestProperty("Accept", "application/json");  
 conn.setRequestProperty("X-M2M-RI", "1234");  
 conn.setRequestProperty("X-M2M-Origin", "CAE1");  
  
 conn.setDoInput(true);  
  
 int responseCode = conn.getResponseCode();  
  
 InputStream is = conn.getInputStream();  
 StringBuilder sb = new StringBuilder();  
 BufferedReader br = new BufferedReader(new InputStreamReader(is, "UTF-8"));  
 String result;  
 while((result = br.readLine())!=null){  
 sb.append(result + "\n");  
 }  
  
 result = sb.toString();  
  
 JSONObject jsonObj = new JSONObject(result);  
 JSONObject m2m = jsonObj.getJSONObject("m2m:cin");  
 JSONObject con = m2m.getJSONObject("con");  
  
 int order = (int)con.get("order");  
 if(order == 2){  
 noti\_order();  
 noti\_by\_tts();  
 timer.cancel();  
 }  
  
 }catch(IOException ie){  
 ie.printStackTrace();  
 }catch(Exception ee){  
 ee.printStackTrace();  
 }  
 }  
 }.start();  
 }  
 };  
 timer.schedule(task, 1000, 5000);  
}

주문 완료를 알리는 noti\_order 코드

public void noti\_order(){  
 NotificationManager notificationManager = (NotificationManager) getSystemService(Context.*NOTIFICATION\_SERVICE*);  
  
 NotificationCompat.Builder builder;  
  
 //Check if Android version is over than Oreo(API26)  
 if(Build.VERSION.*SDK\_INT* >= Build.VERSION\_CODES.*O*){  
 String channelID = "channel\_01";  
 String channelName = "MyChannel01";  
  
 NotificationChannel channel = new NotificationChannel(channelID, channelName, NotificationManager.*IMPORTANCE\_DEFAULT*);  
  
 notificationManager.createNotificationChannel(channel);  
  
 builder = new NotificationCompat.Builder(this, channelID);  
 }  
 else{  
 builder = new NotificationCompat.Builder(this, "default");  
 }  
 //Setting notification alarm's icon  
 builder.setSmallIcon(R.drawable.*ic\_launcher\_foreground*);  
  
 //Setting alarm's title and text  
 builder.setContentTitle("Order App");  
 builder.setContentText("Your order is completed.");  
  
 //If alarm is clicked, then alarm would be destroyed.  
 builder.setAutoCancel(true);  
 //Setting alarm vibration  
 builder.setVibrate(new long[]{0, 2000, 1000, 3000});  
  
 Notification notification = builder.build();  
  
 notificationManager.notify(1, notification);  
}

주문 완료를 TTS 서비스로 알리는 noti\_by\_tts 코드

public void noti\_by\_tts(){  
 String text = "주문이 완료되었습니다.";  
 if(Build.VERSION.*SDK\_INT* >= Build.VERSION\_CODES.*LOLLIPOP*){  
 tts.speak(text, TextToSpeech.*QUEUE\_FLUSH*, null, null);  
 }  
 else{  
 tts.speak(text, TextToSpeech.*QUEUE\_FLUSH*, null);  
 }  
}