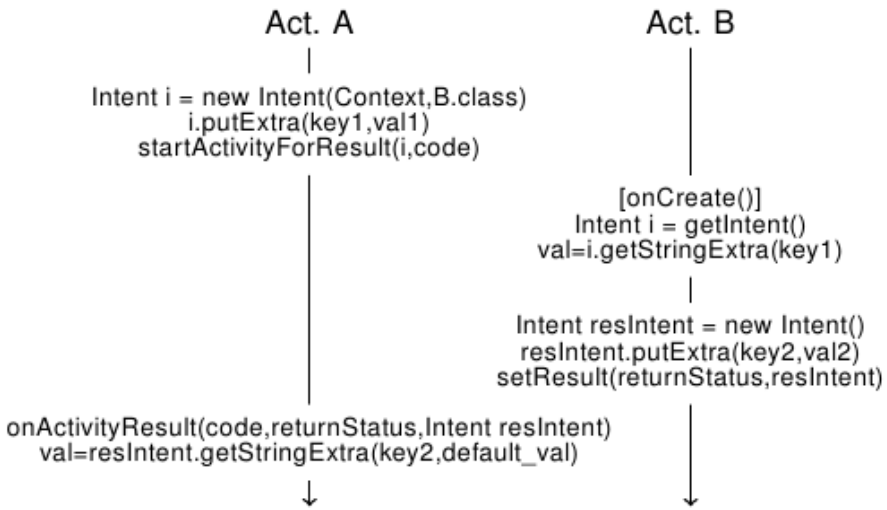
Les Intents



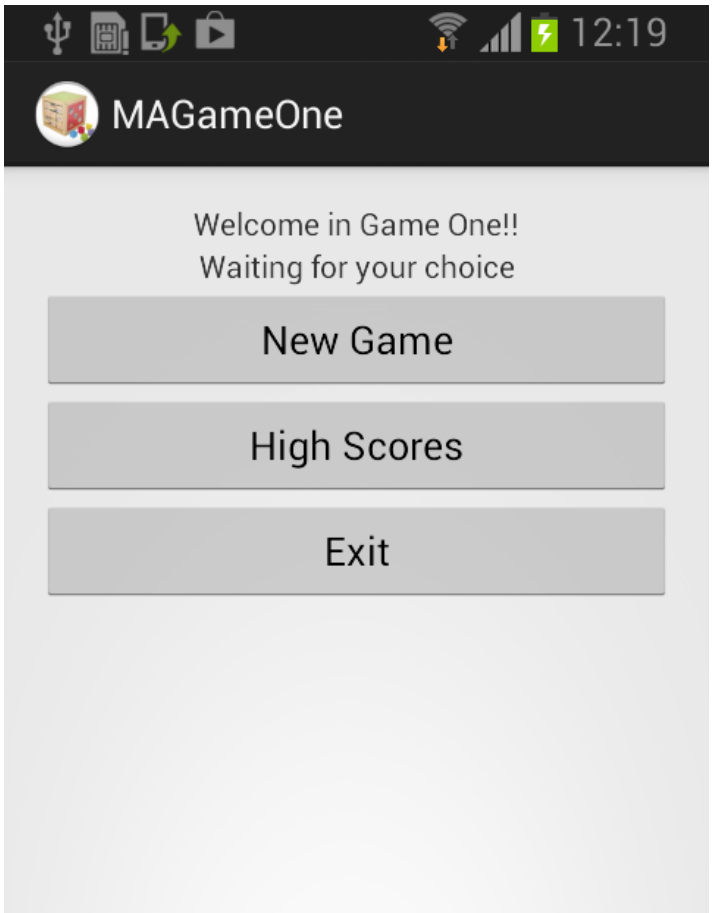
IUT département informatique - Université du Havre

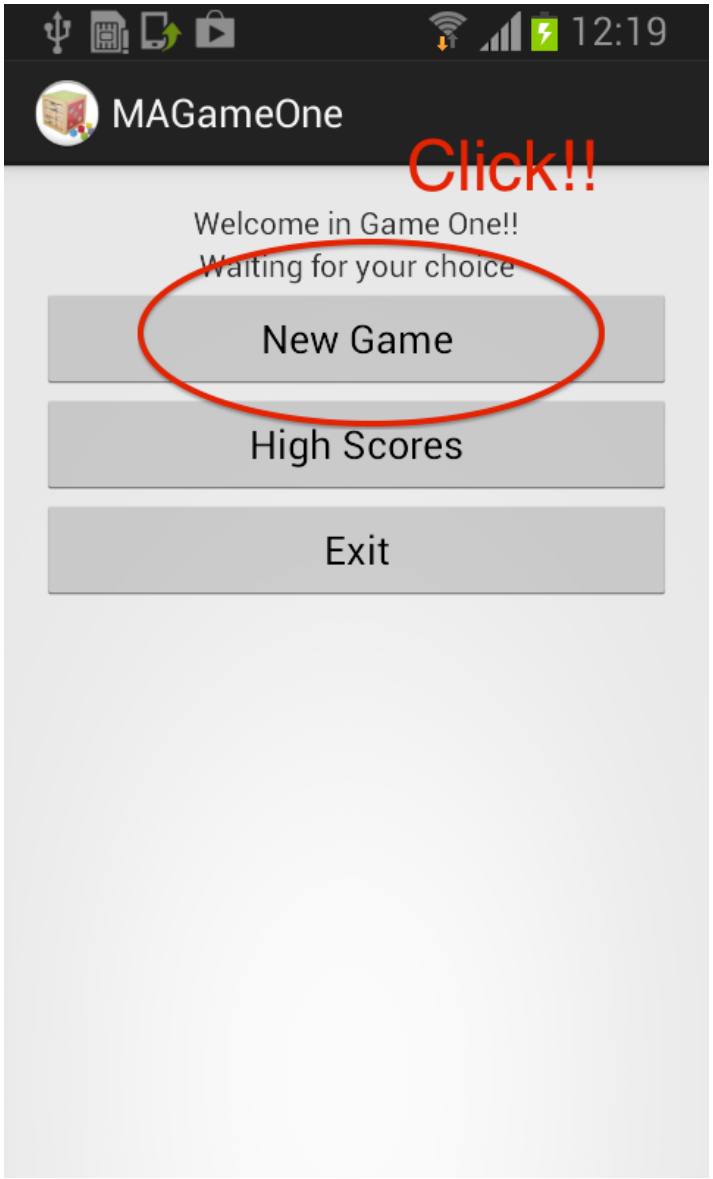
Plan

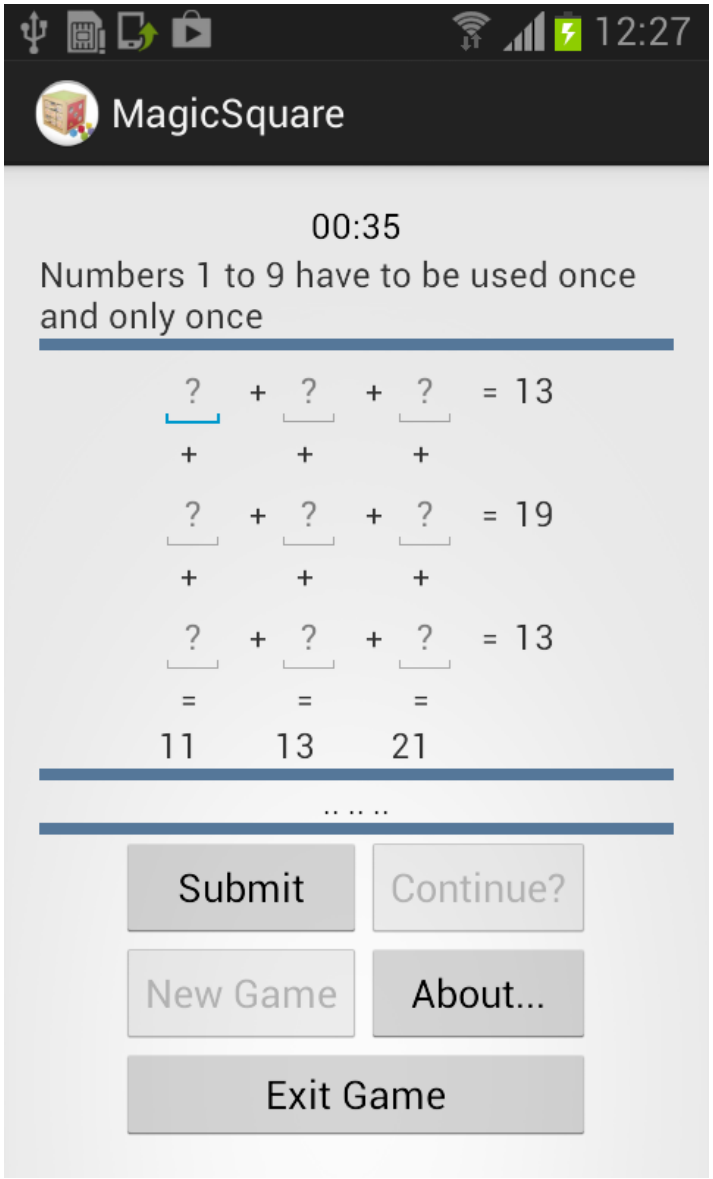
- introduction
- Intents
  - de quoi s'agit-il ?
  - comment construire un Intent
  - comment utiliser les Intents pour transmettre des informations entre composants ?
- résumé
- des questions ?

Exemple illustratif

- une app proposant plusieurs jeux
- architecture : une page d'accueil générale et une page d'accueil pour chaque jeu
- chaque jeu est lancé depuis la page d'accueil




12:19
MAGameOne
Click!!
Welcome in Game One!!
Waiting for your choice
New Game
High Scores
Exit

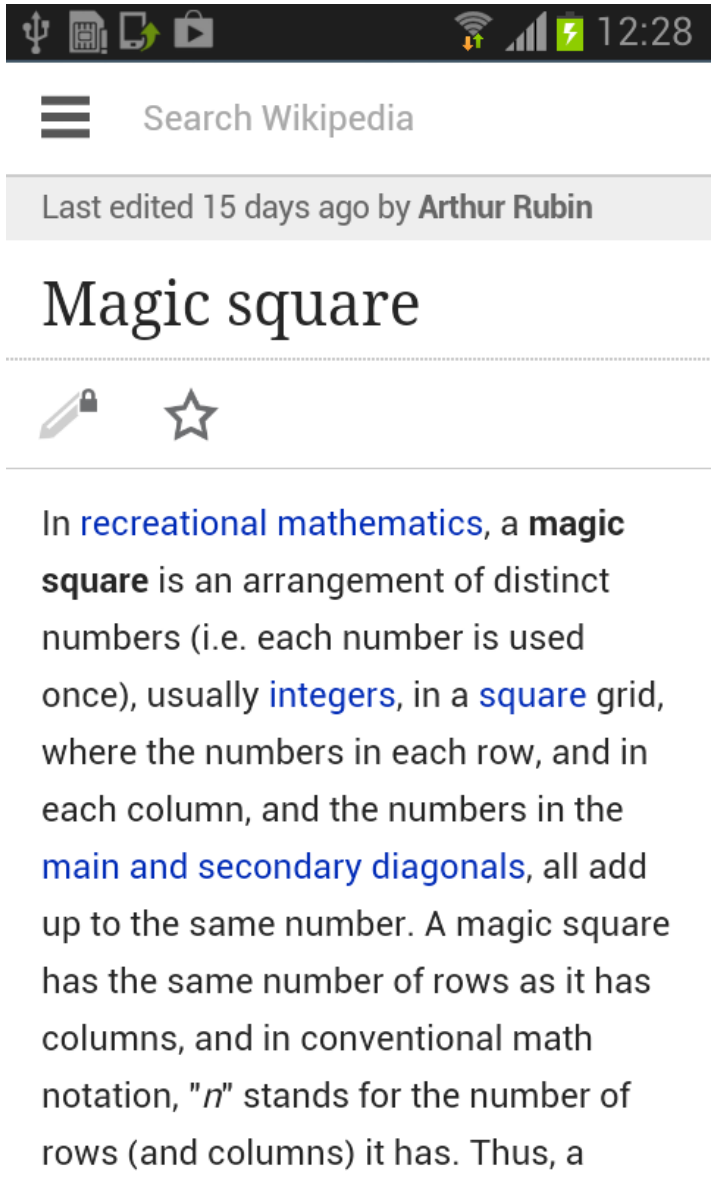

12:27
MagicSquare
00:35
Numbers 1 to 9 have to be used once and only once
? + ? + ? = 13
+ + +
? + ? + ? = 19
+ + +
? + ? + ? = 13
= = =
11 13 21
.. .. ..
Submit
Continue?
New Game
About...
Exit Game

règles du jeu

- au cours du jeu, ou avant de commencer, l'utilisateur désire avoir des informations complémentaires sur le jeu (règles, origines, etc.)
- il choisit de cliquer sur le bouton "About..." ce qui déclenche...

  l'ouverture d'une page web


12:27
MagicSquare
00:35
Numbers 1 to 9 have to be used once and only once
? + ? + ? = 13
+ + +
? + ? + ? = 19
+ + +
? + ? + ? = 13
= = =
11 13 21
.. .. ..
Submit
Click !!!
New Game
About...
Exit Game

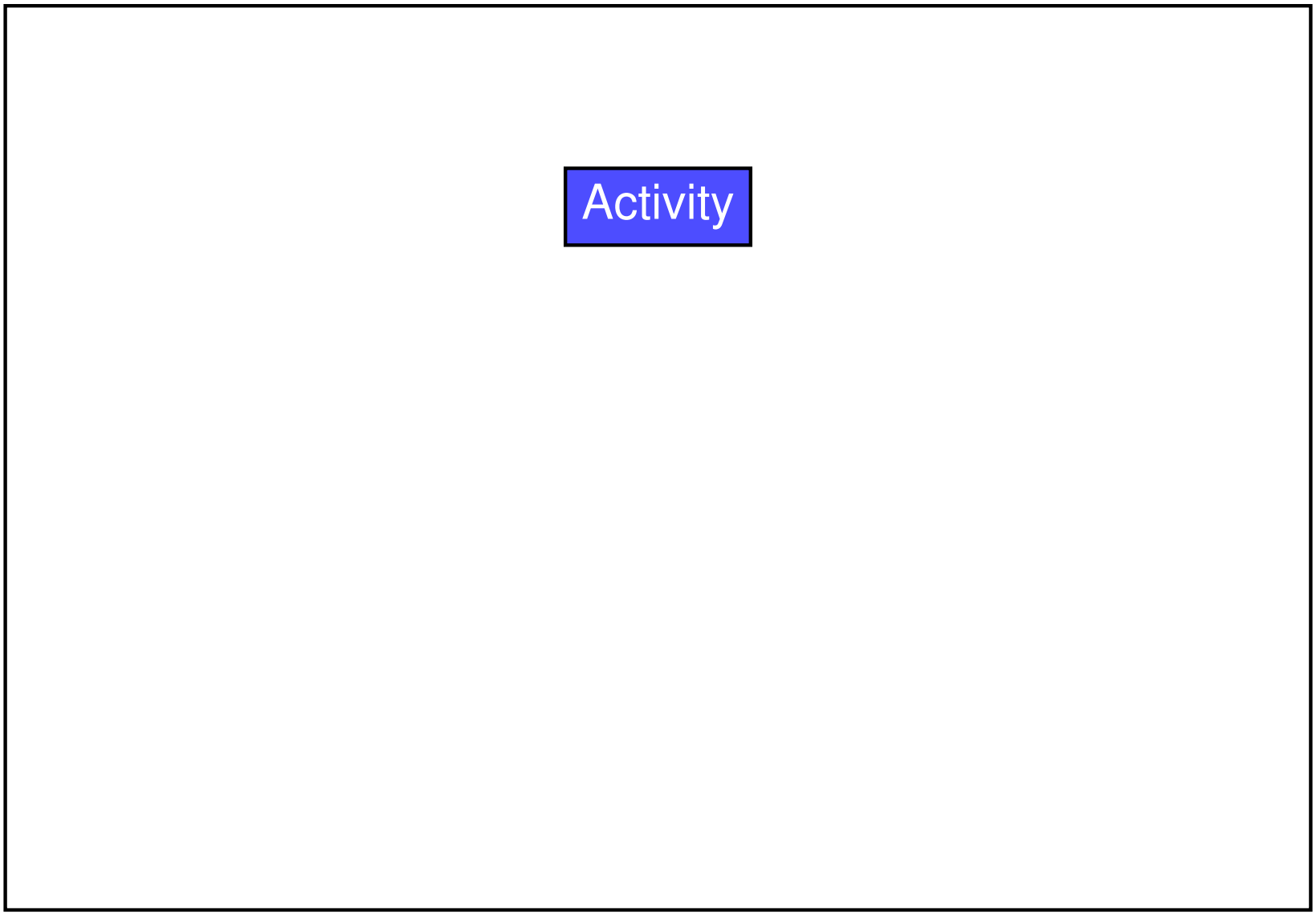

12:28
Search Wikipedia
Last edited 15 days ago by Arthur Rubin
Magic square
In recreational mathematics, a magic square is an arrangement of distinct numbers (i.e. each number is used once), usually integers, in a square grid, where the numbers in each row, and in each column, and the numbers in the main and secondary diagonals, all add up to the same number. A magic square has the same number of rows as it has columns, and in conventional math notation, "n" stands for the number of rows (and columns) it has. Thus, a

Intent ?

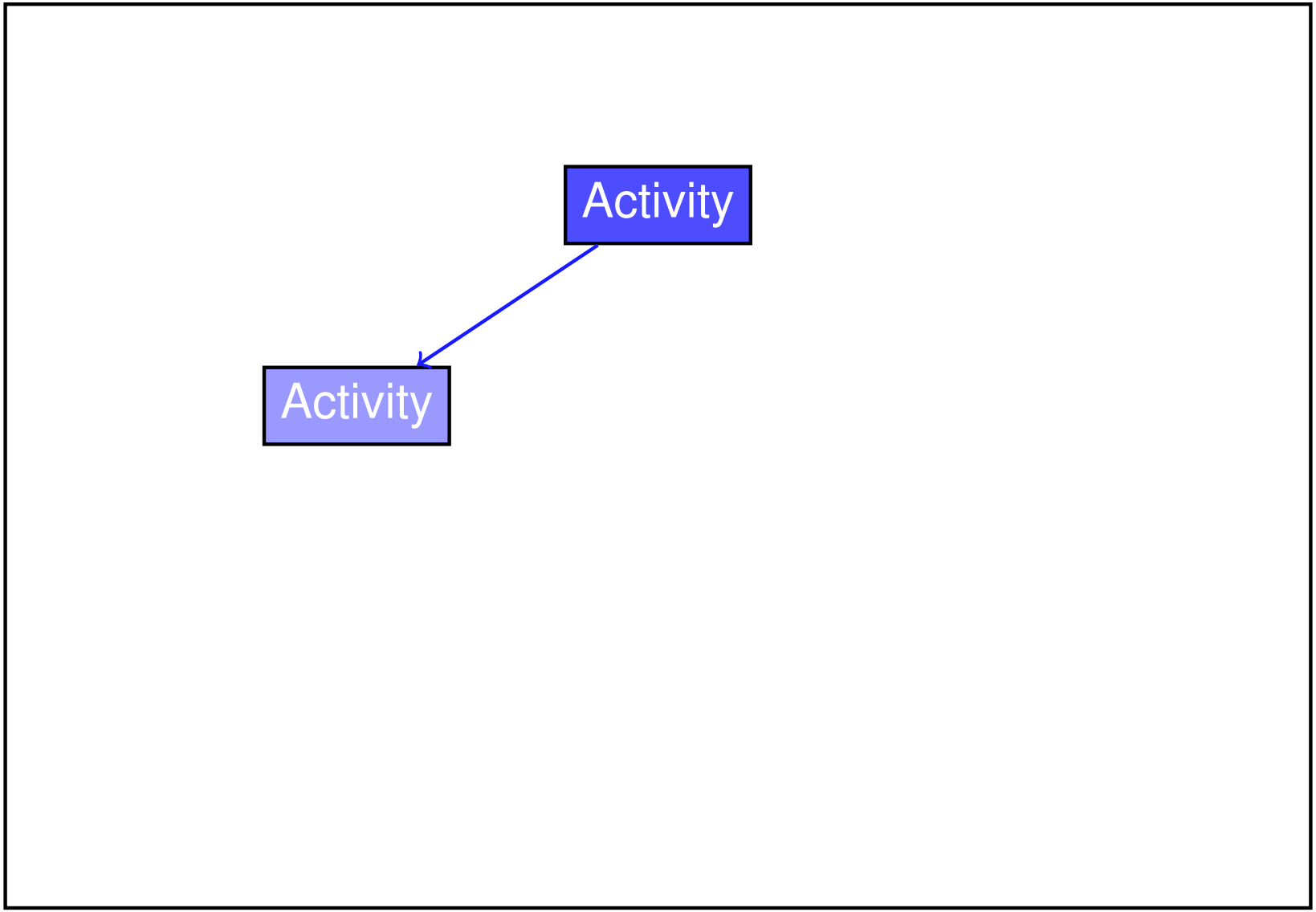
- chacune de ces opérations est le résultat d'un **Intent**.
- l'un des élément clef en programmation Android
- les Intents permettent le démarrage de certains composants depuis d'autres composants
- l'intent peut éventuellement transporter des données
- lors de la fin d'une Activity l'intent peut également envoyer des données qui seront transmises à l'Activity appelante

- un Intent est une description d'une action à effectuer
- ses deux principaux *paramètres* sont :
  - l'action à effectuer
  - les données qui l'accompagnent

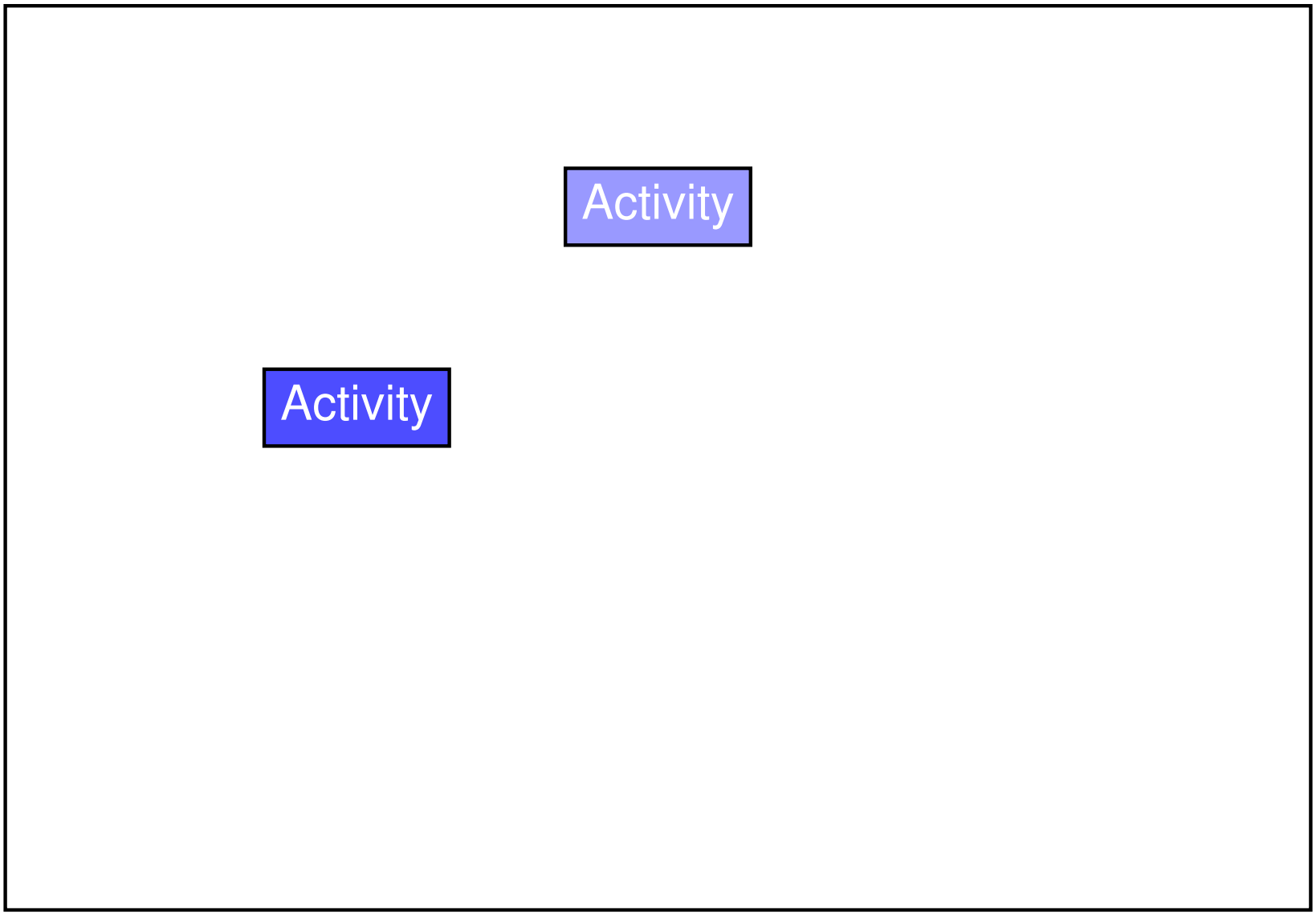
Intents



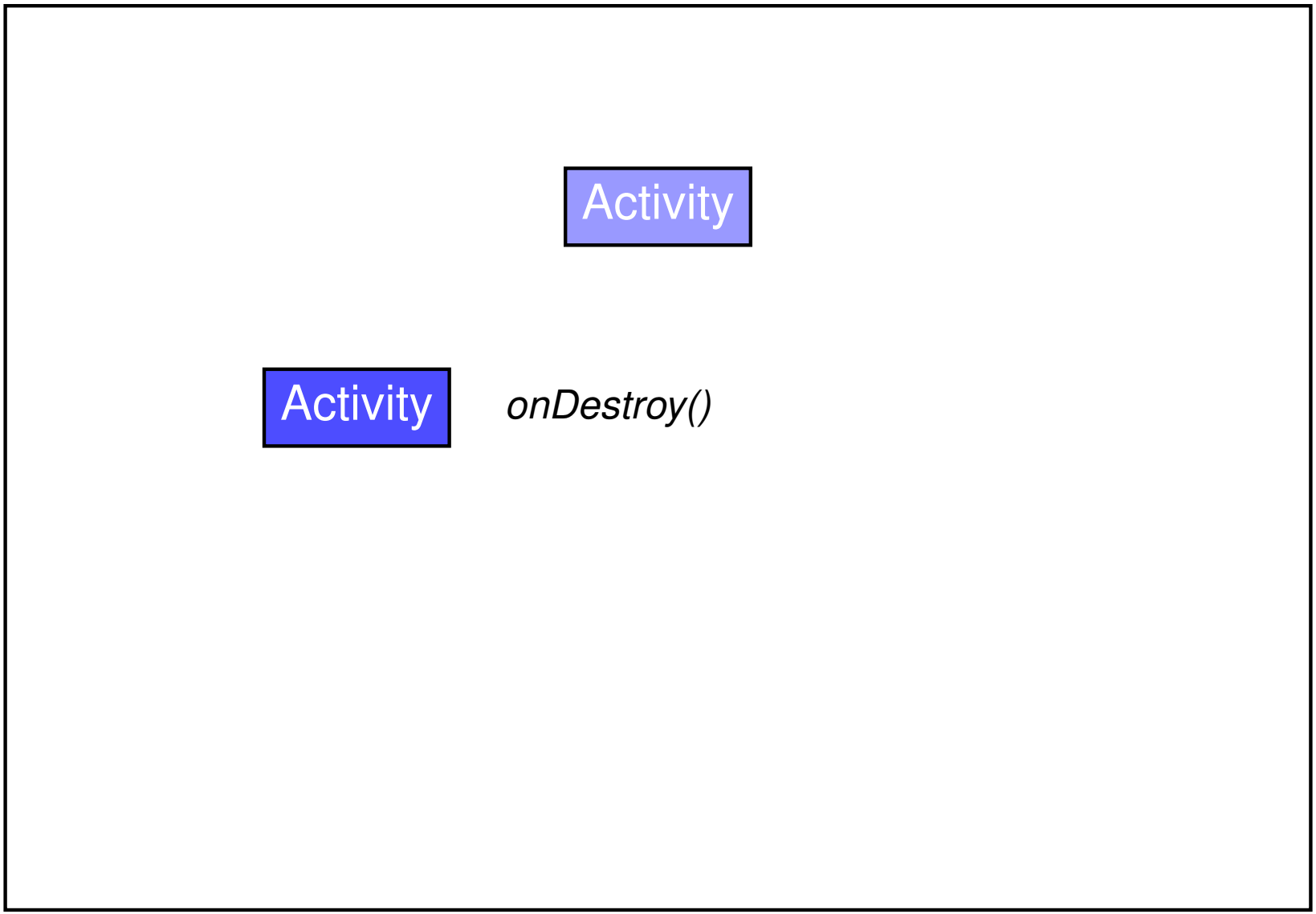
Intents



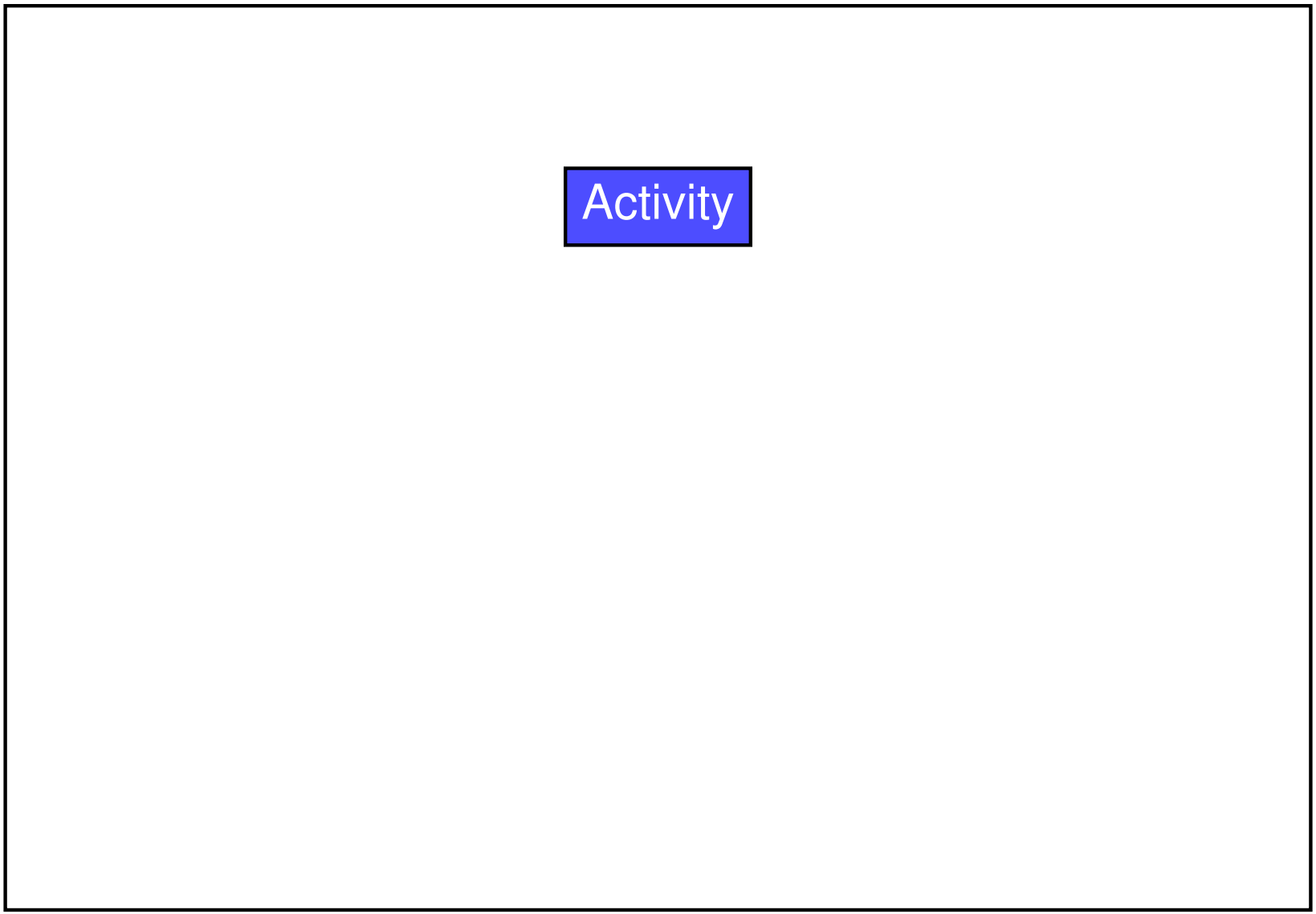
Intents



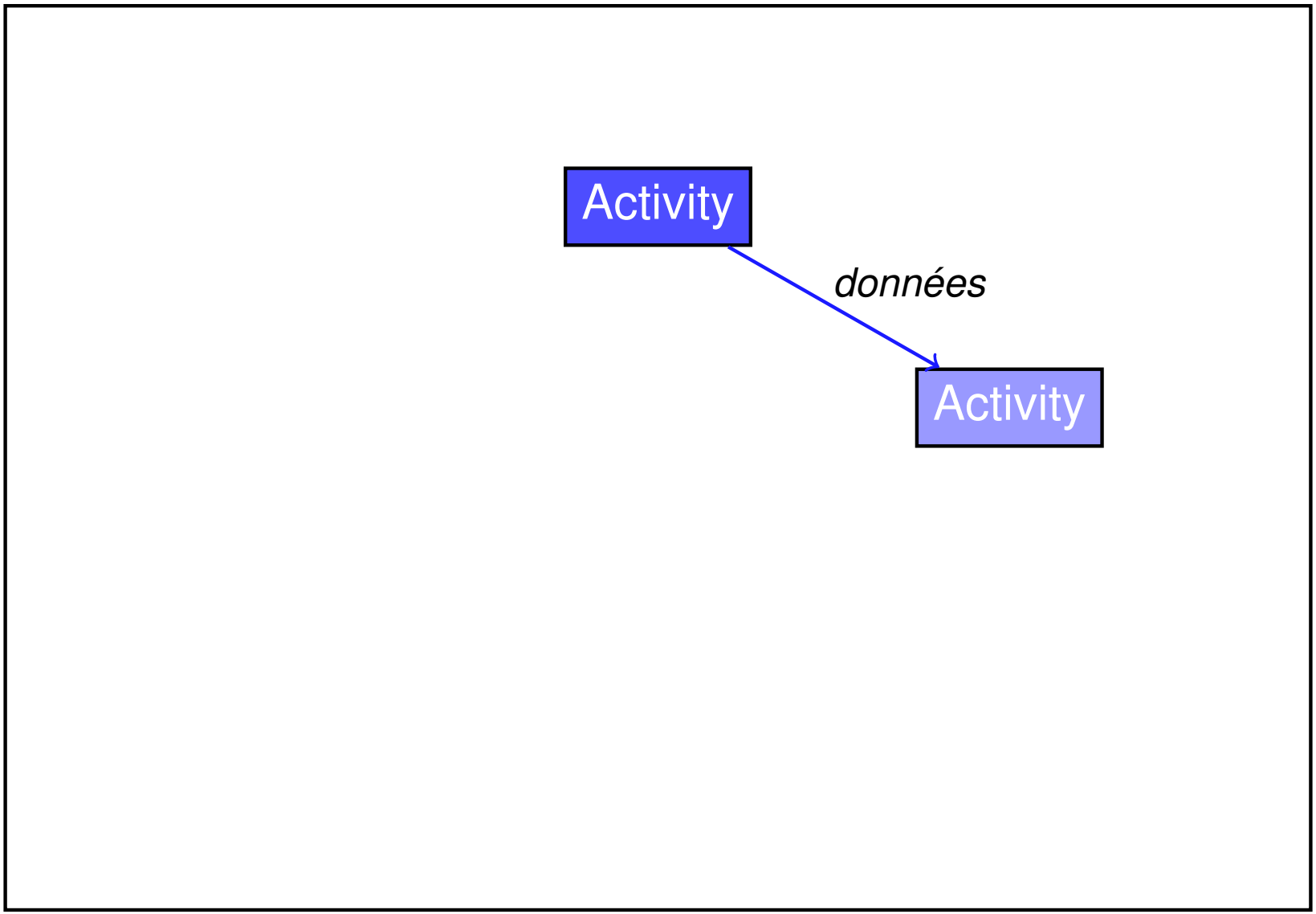
Intents



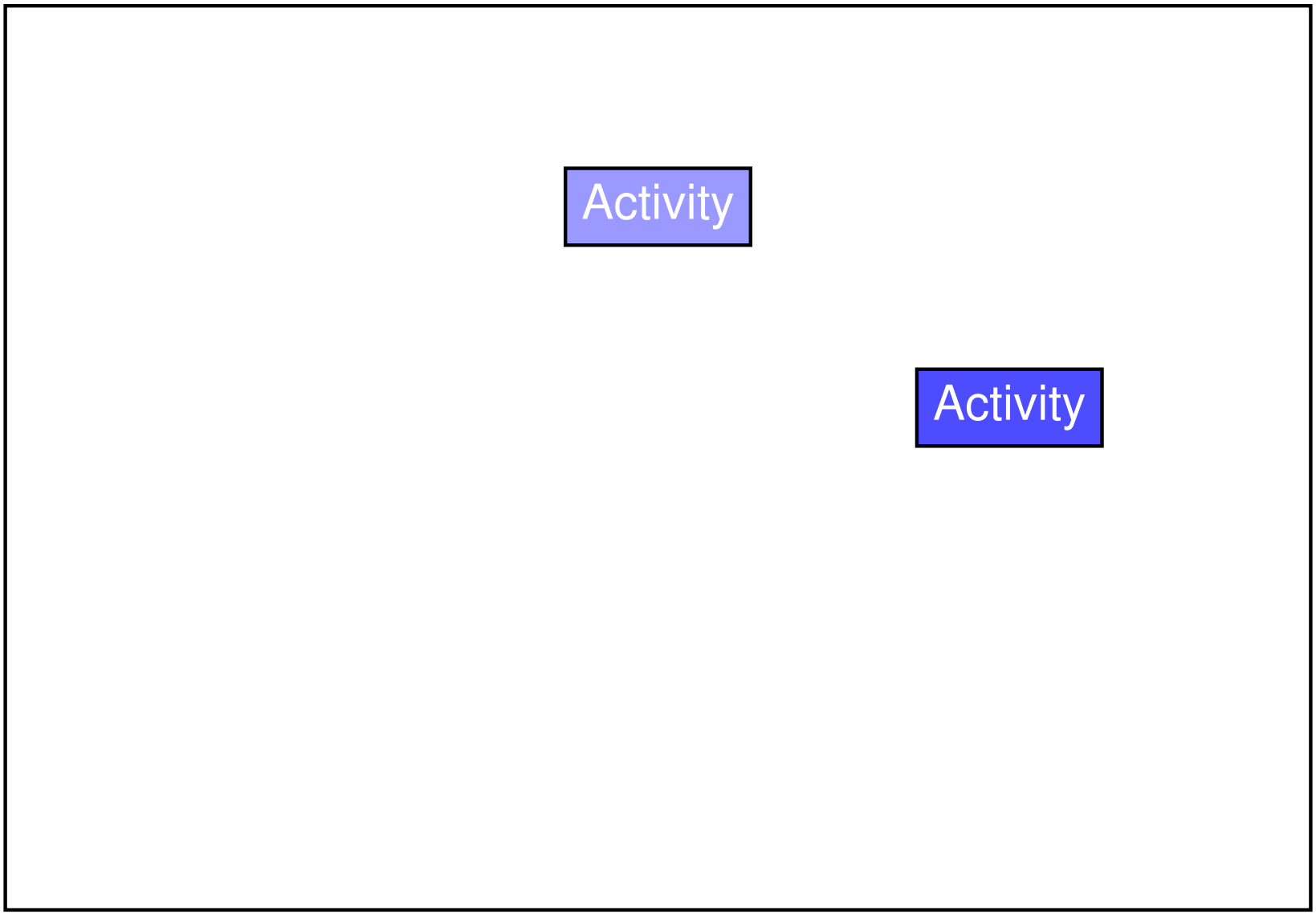
Intents



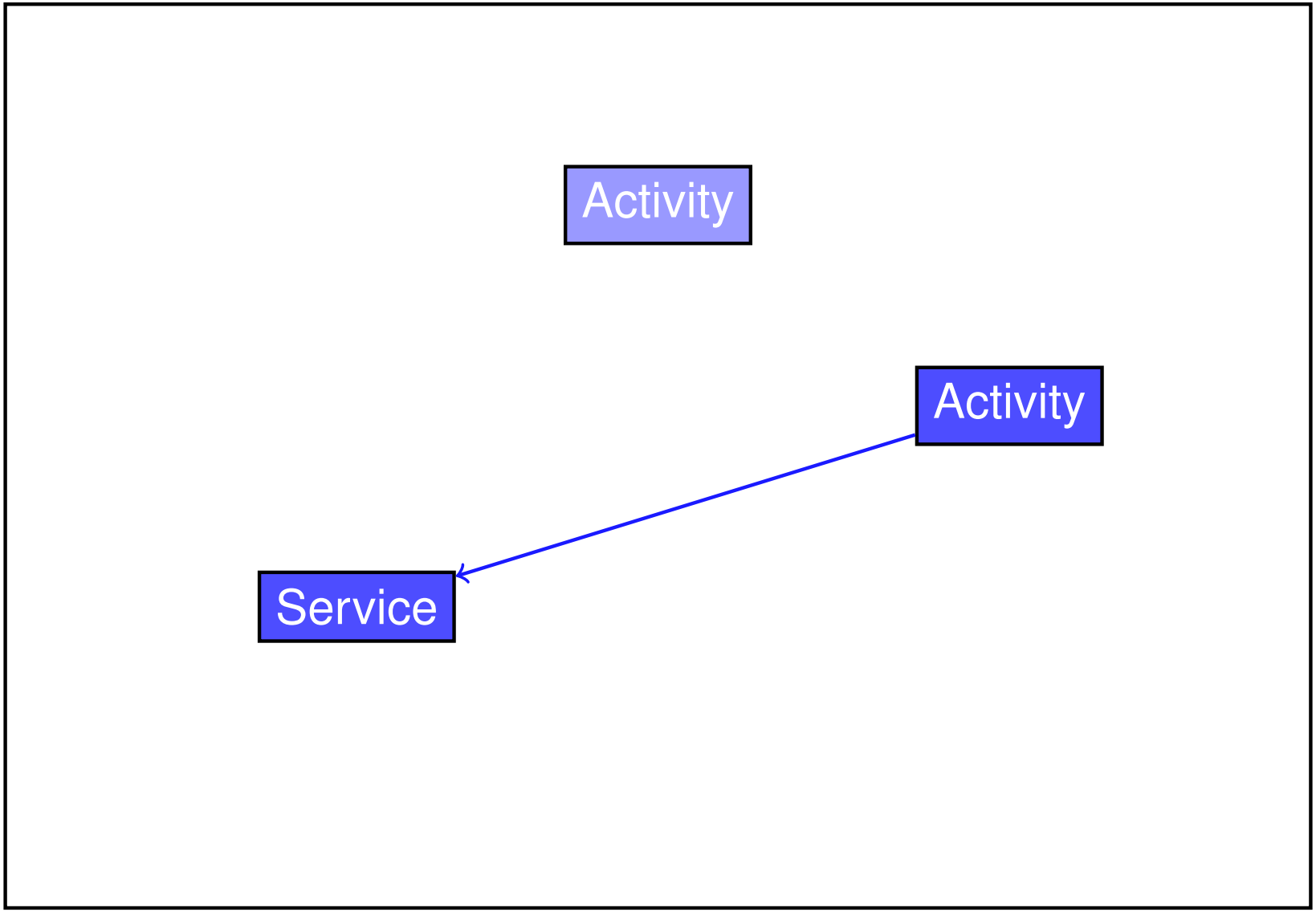
Intents



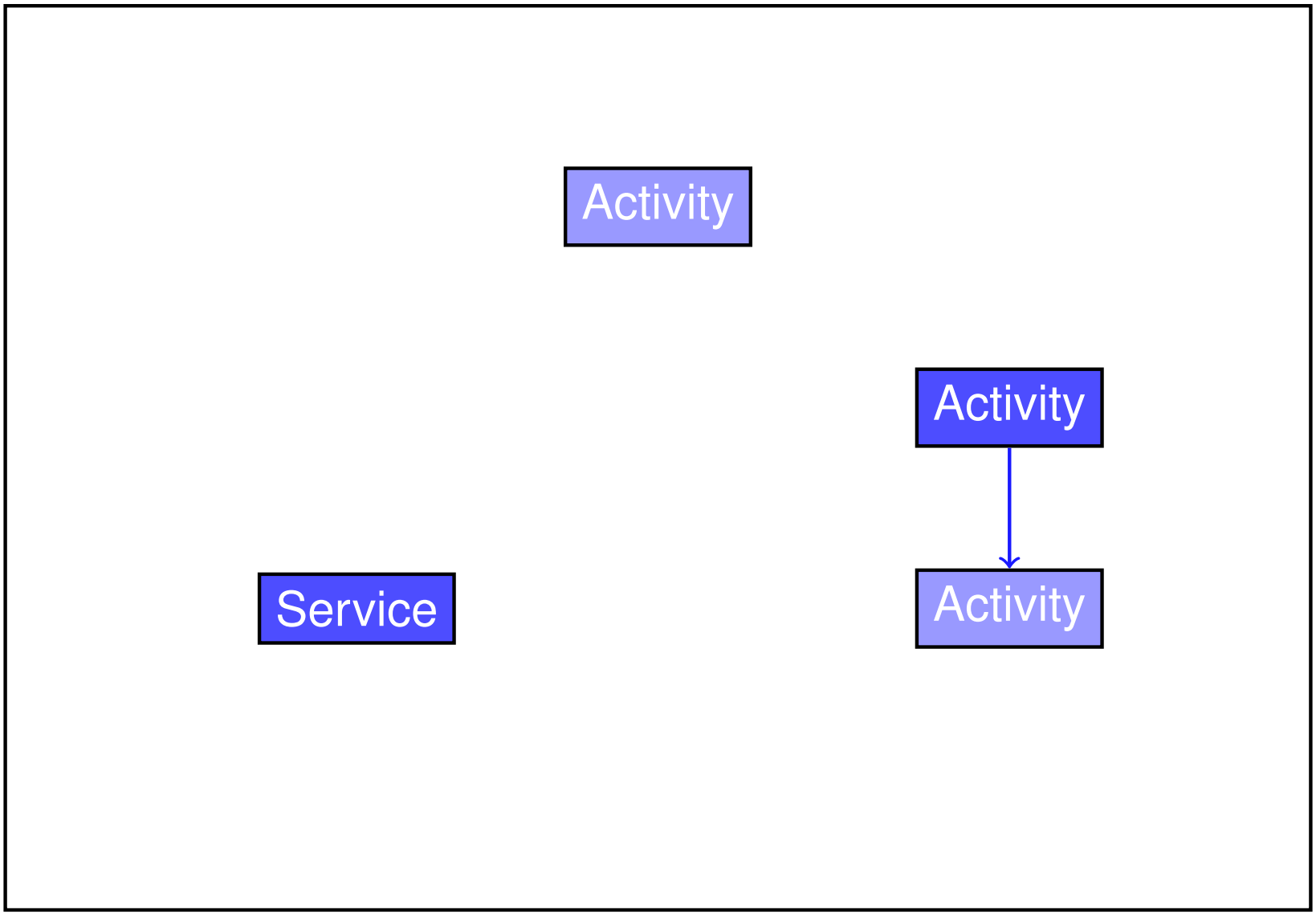
Intents



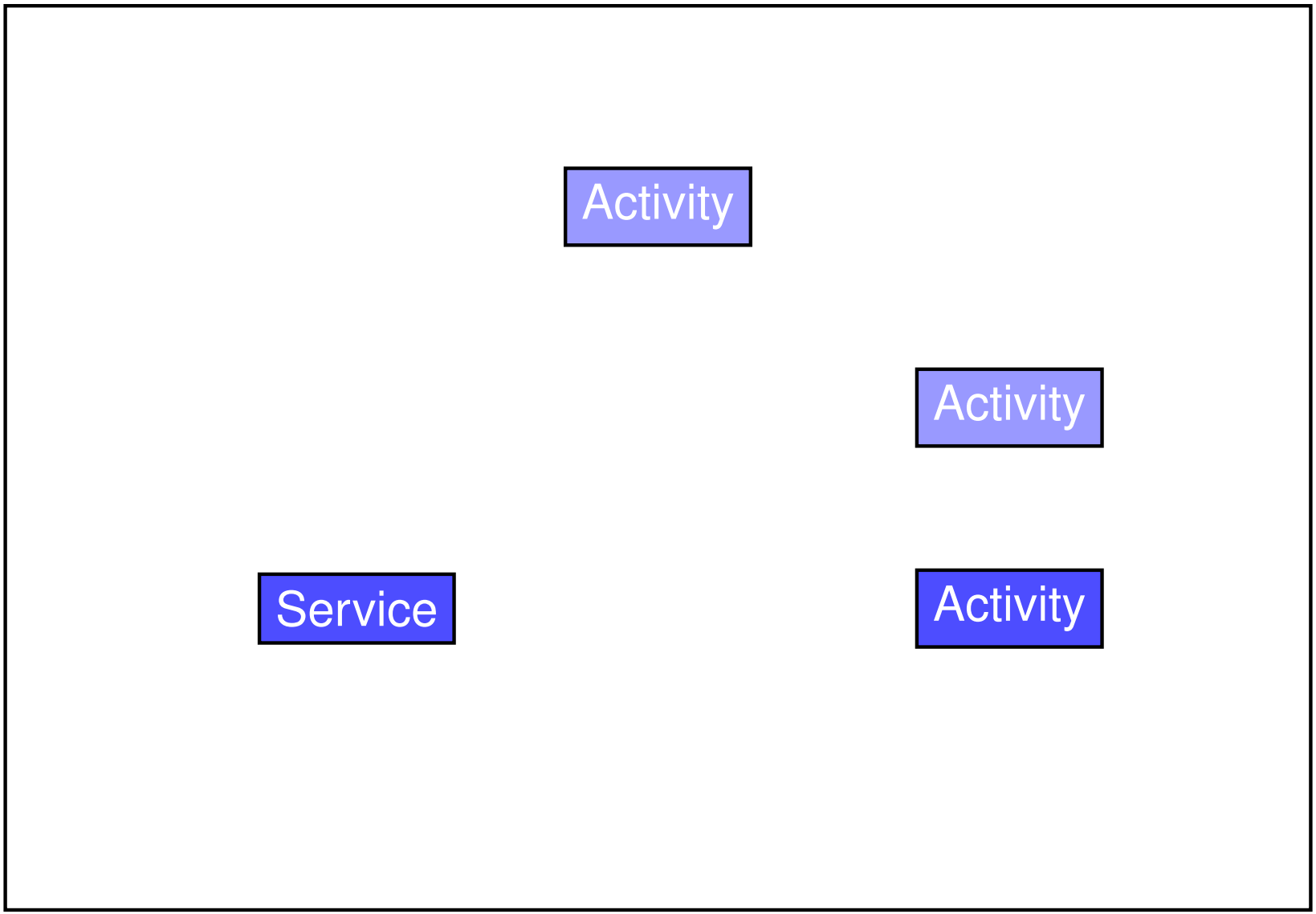
Intents



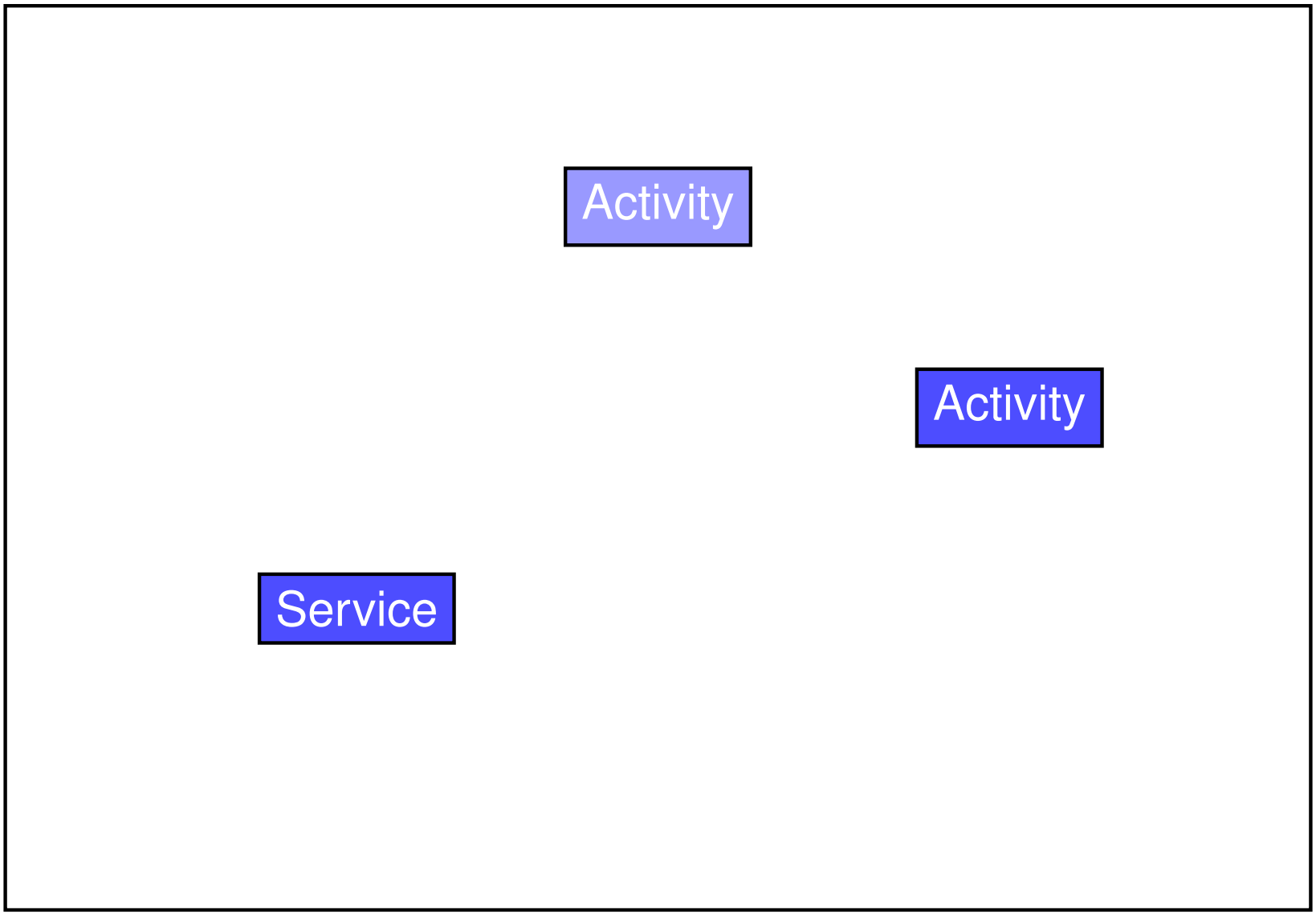
Intents



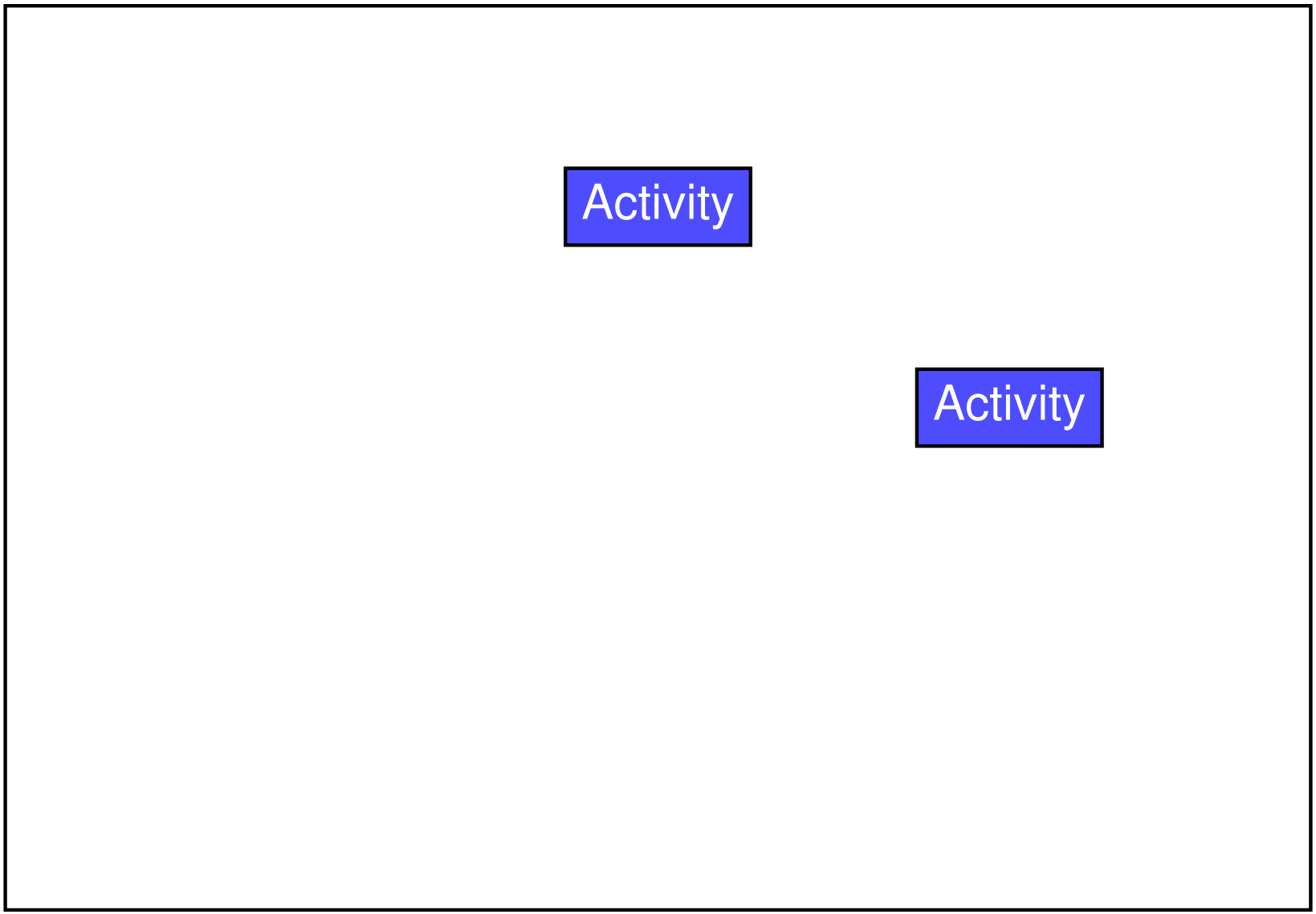
Intents



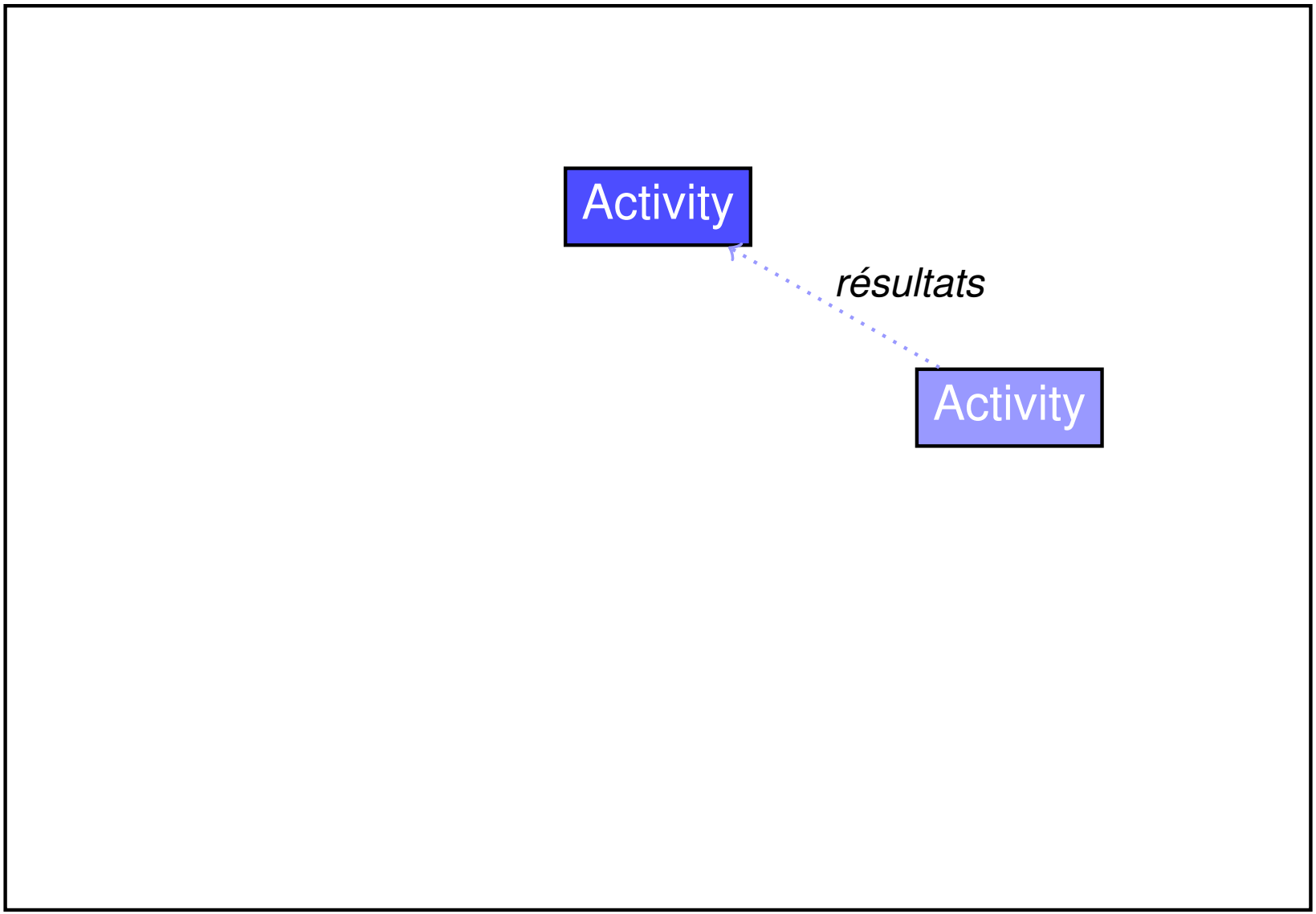
Intents



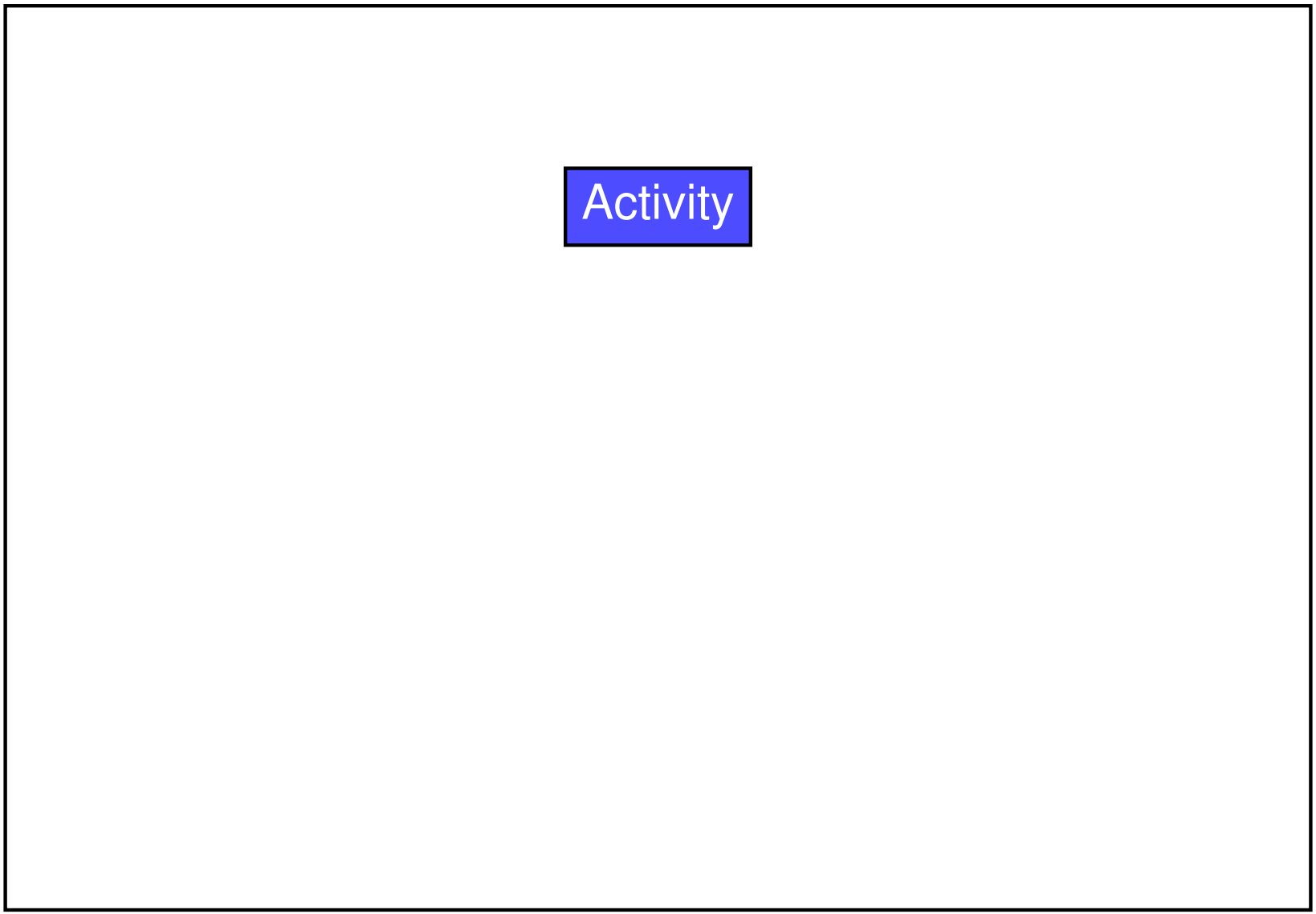
Intents



Intents

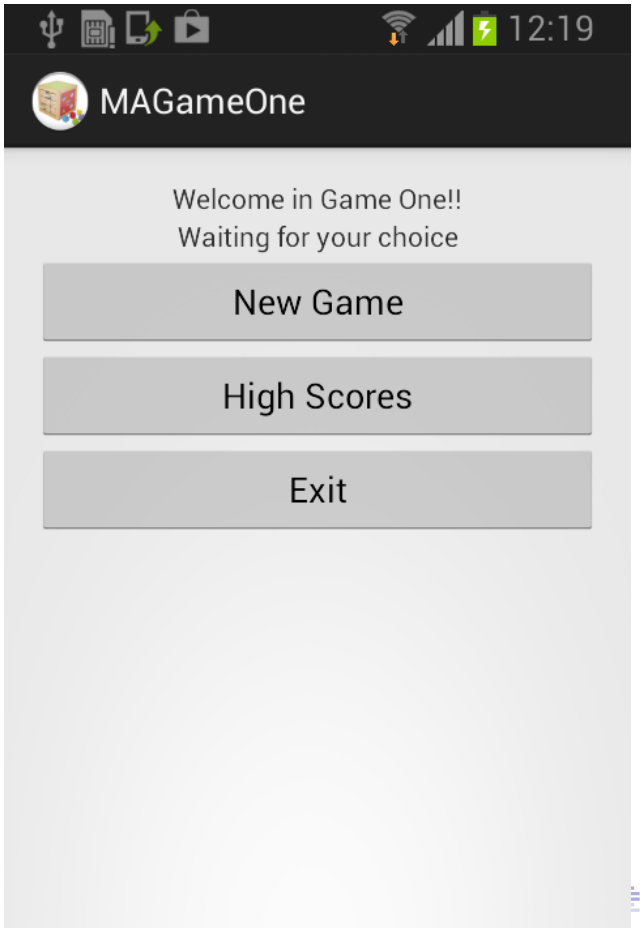


Intents



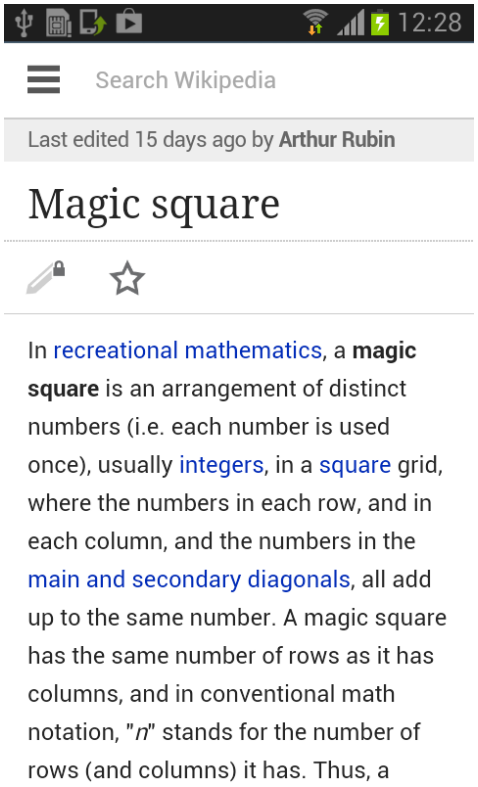
dans quel but sont-ils utilisés ?

- démarrage d'une app/activity/service connaissant sa Classe
  → e.g. démarrer une occurrence de mobapp.multiactivity.MagicSquare
- intent explicit



dans quel but sont-ils utilisés ?

- démarrage d'une app/activity/service connaissant l'action que ce composant doit être capable d'exécuter
  → e.g. démarrer une app capable d'afficher une page web
- intent implicite



dans quel but sont-ils utilisés ?

- démarrer un service (qui s'exécute en tâche de fond)
  $\rightarrow$ e.g. une app de lecture de musique
- le démarrage d'un service se fait préférentiellement de manière explicite

dans quel but sont-ils utilisés ?

- diffusion d'information pour avertir d'autres composants de la survenue d'un événement particulier
  $\rightarrow$ e.g. informer les utilisateurs qu'ils entrent dans une zone à risque

exemples d'utilisation

- Cas 1 : démarrer un service ou une app précise
- Cas 2 : démarrer une app ou une activity en fonction de l'action attendue sur une donnée particulière
- Cas 3 : diffuser des informations sur un événement. Dans ce cas, l'Intent est invisible pour l'utilisateur, seuls les Broadcast receiver les détectent

description des Intents

- un Intent est caractérisé par plusieurs champs
- en fonction du type d'Intent différentes informations sont utilisées pour sa création

description

Explicite ↔ Implicite

- si le nom du composant est donné
  → Intent Explicite
- le nom du composant peut être passé en paramètre au constructeur ou précisé par l'appel à l'une des méthodes suivantes : setComponent(), setClass() or setClassName()
- il faut utiliser le nom qualifié complet *mobapp.multiapp.TheActivity*

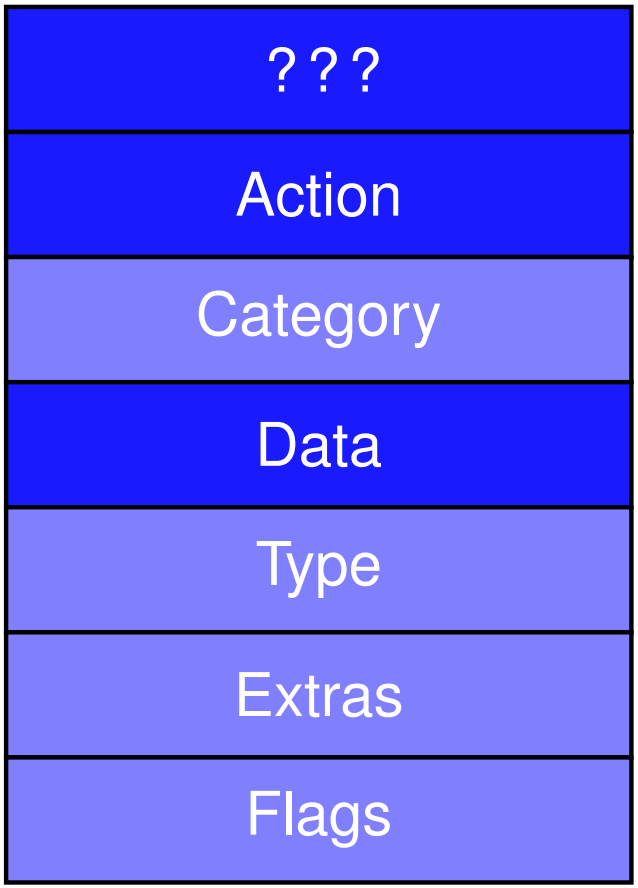
description

Explicite $\leftrightarrow$ Implicite

- si le nom de la classe n'est pas indiqué

  $\rightarrow$ Intent Implicite

description

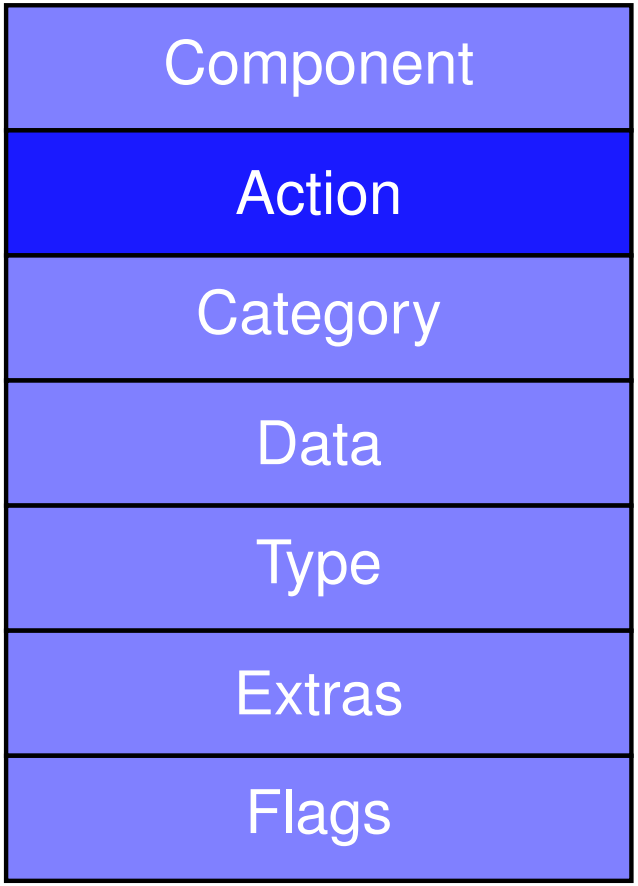


Explicite ↔ Implicite

- si le nom de la classe n'est pas indiqué

  → Intent Implicite
- dans ce cas, Action et Data sont requis

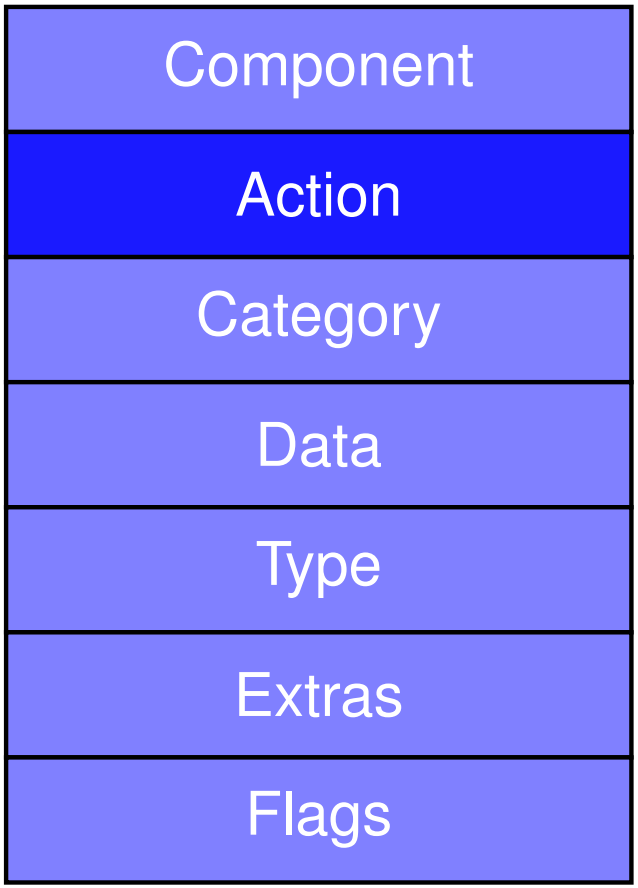
description



Action

- indique ce que le composant est censé faire
- il existe une grande quantités d'Action Android natives :
  - ACTION_MAIN pour démarrer une nouvelle activity (point d'entrée)
  - ACTION_DIAL pour composer un numéro de téléphone
  - ACTION_SENDTO pour envoyer un nouveau message
  - ACTION_VIEW pour afficher des données

description



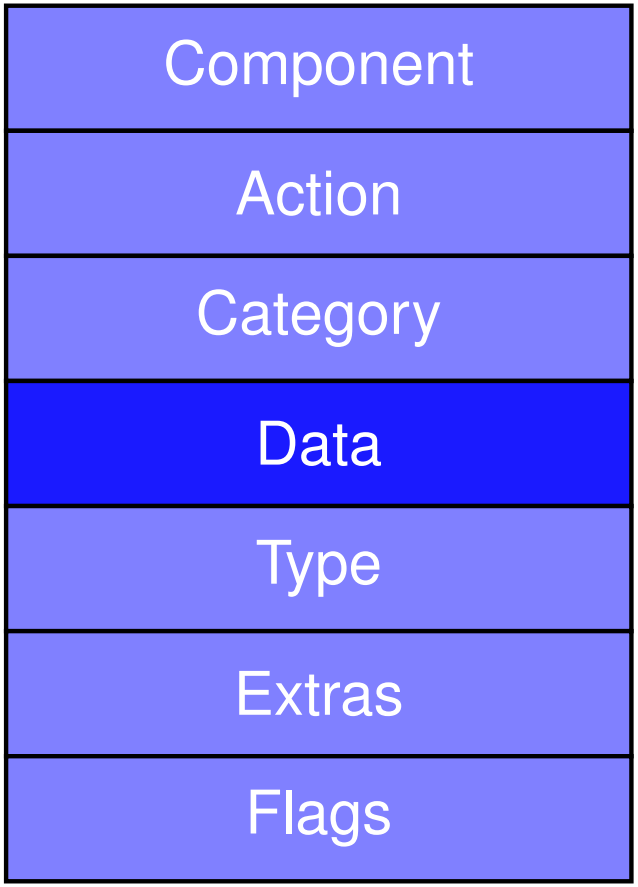
Action

- l'Action peut être spécifiée avec setAction() ou peut être ajouter comme un des paramètres du constructeur

Action ACTION_VIEW

- très souvent utilisé
- pour l'affichage des pages web, des photos, de lieux, etc. en fonction des données transmises

description



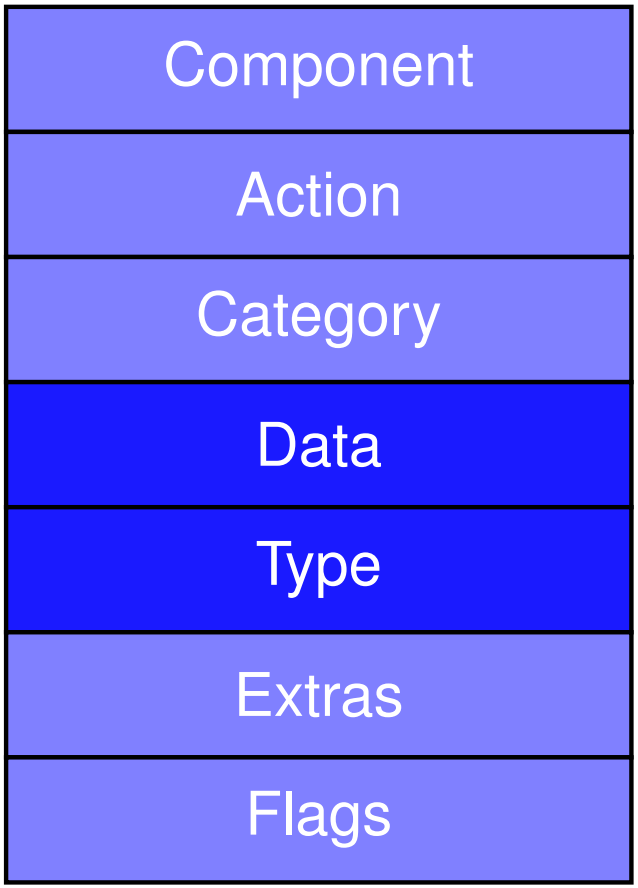
Data

- il s'agit de *Uniform Resource Identifier* (URI). Format :
- scheme :info{ ?query}{#frag}
- exemple
  http ://duckduckgo.com/ ?q=android

construire des URI

- Uri webpage =
  Uri.parse("http ://duckduckgo.com/ ?android") ;
- Uri tel = Uri.parse("tel :+336123456789") ;
- Uri geo =
  Uri.parse("geo :52.312405,20.918330") ;

description



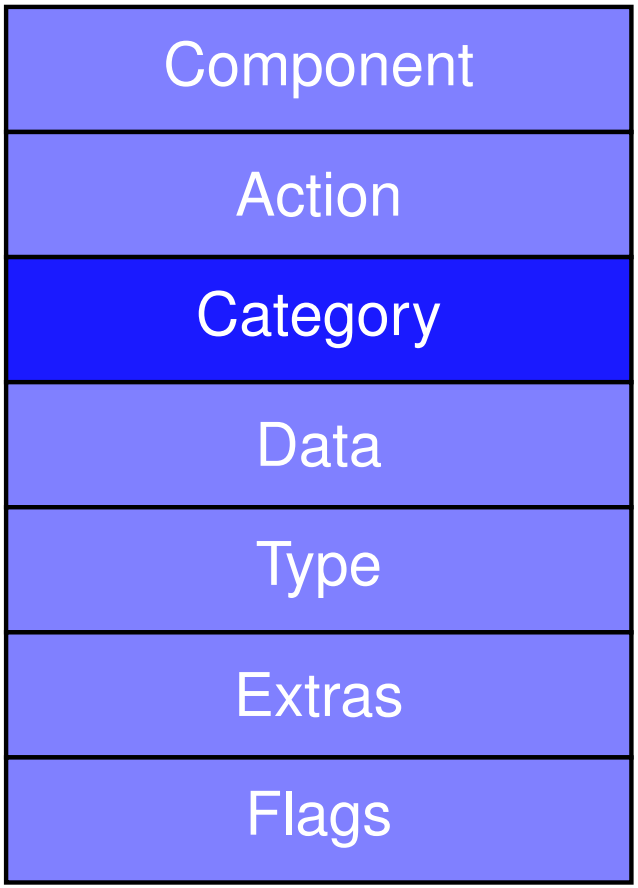
URI

- Uri webpage = Uri.parse("http ://duckduckgo.com/ ?android") ;
- Uri tel = Uri.parse("tel :+336123456789") ;
- Uri geo = Uri.parse("geo :52.312405,20.918330") ;

Type

- le type de la donnée peut être ajouté aux données elles-mêmes pour aider Android à trouver les app les plus pertinentes pour les traiter
- MIME Type : toujours en minuscules
- text/html video/avi application/msword application/x-gzip image/jpeg audio/x-mpeg-3 video/mpeg application/pdf image/png etc.

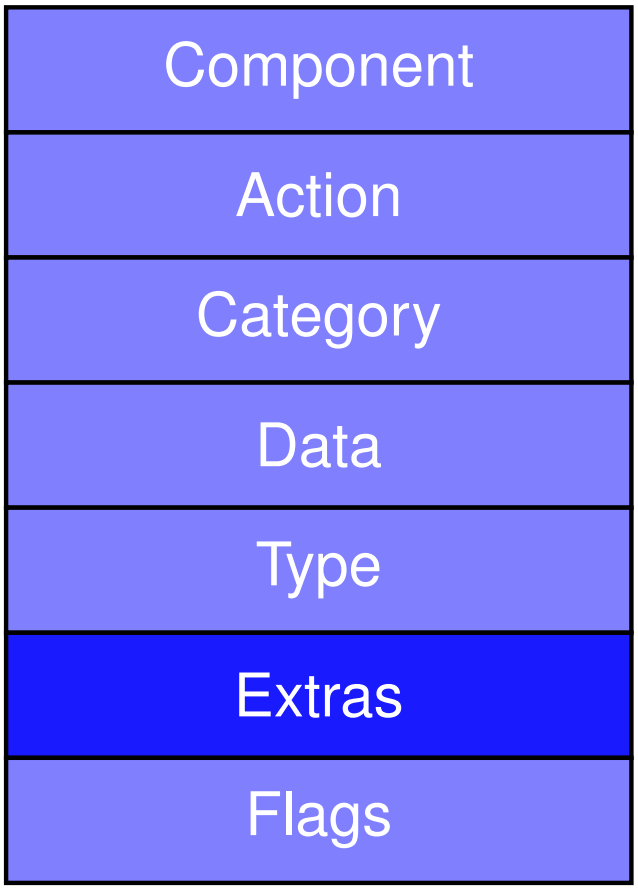
description



Category

- information additionnelle concernant le composant recherché pour répondre à l'Intent
- la plupart des Intent ne nécessite pas de category, à l'exception de CATEGORY_LAUNCHER

description



Extra

- données additionnelles
- données stockées et transmises sous la forme de quelques couples (clef, valeur) et ajouté en utilisant la méthode putExtra()
- il est possible d'ajouter un Bundle et de l'ajouter avec la méthode putExtras()
- certaines clefs existent déjà (e.g. EXTRA_MAIL) pour les types de données standardisés

description

Flags

- permettent de contrôler la manière dont l'intent sera considéré
- exemple : FLAG_ACTIVITY_LAUNCH_ADJACENT est utilisé pour démarrer l'activité à côté de l'activité appelante dans le contexte du mode *split-screen multi-window*
- exemple : FLAG_ACTIVITY_CLEAR_TOP. Si la back stack est composée des activités A, B, C et D (D est en haut de la pile), si B est démarré avec ce flag dans l'intent alors C et D seront terminées et la pile sera maintenant A, B.
- remarque : nous ne les utiliserons pas.

Construire des Intents

comment construire un Intent

Intent explicite - Démarrage d'une nouvelle Activity

- le constructeur est Intent myIntent = new Intent(Context,Class) ; pour un Intent explicite
- le context est transmis par l'Activity courante (on utilise souvent *this*
- le paramètre *Class* est le nom qualifié de l'Activity que l'on souhaite démarrer : mobapp.testintent.ExplicitActivity (sauf si l'Activity appartient au même package)

```
Intent myIntent = new Intent(this,ExplicitActivity.class);
startActivity(myIntent);
```

comment construire un Intent

Intent explicite - démarrage avec envoi de données

- on peut compléter l'Intent en lui ajoutant des données
- c'est réalisé avec la méthode putExtra()
- les données sont ajoutées selon le format habituel : un couple (clef,valeur)
- de préférence déclarez les clefs comme des public final static String

```
public final static String URL = "mobapp.testintent.URL";

// ...some piece of code...

Intent myIntent = new Intent(this,ExplicitActivity.class);
myIntent.putExtra(HomeActivity.URL,url);
startActivity(myIntent);
```

Intent explicite - côté Activity démarrée

- si aucune données, rien à faire
- si des données accompagnent l'Intent, dans la code de *onCreate()* il faut récupérér l'Intent avec la méthode getIntent()
- pour récupérer les données, il est nécessaire d'avoir la clef : get**String**Extra(clef)

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_explicit);
    Intent myIntent = getIntent();
    url = myIntent.getStringExtra(HomeActivity.URL);
    ((TextView)findViewById(R.id.datareceived)).setText(url);
}
```

Intents implicites

comment construire un Intent

Intent implicite

- nous souhaitons afficher une page web avec un browser existant (peu importe lequel)
- dans ce cas nous pouvons construire un Intent implicite dans lequel le nom de la classe ne va pas apparaître mais l'action qui devra être effectuée par ce composant

```
public void start(View view) {
    Intent webIntent = new Intent(Intent.ACTION_VIEW,Uri.parse(url));
    startActivity(webIntent);
}
```

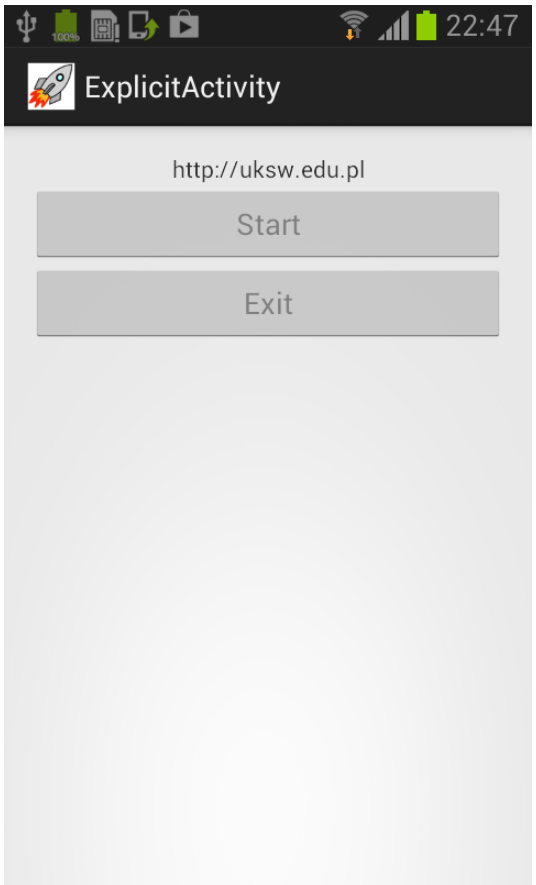
dans le cas des Intent implicites, avant l'appel nous ne savons pas quelle app va se charger d'effectuer l'action demandée

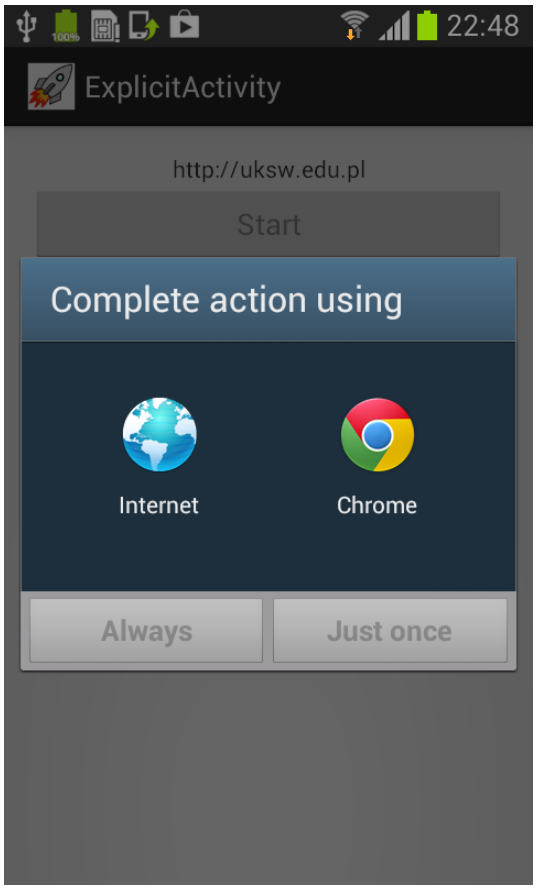
comment construire un Intent

Intent implicite

- le mécanisme en charge de trouver toutes les app aptes à exécuter l'action requise est appelé intention resolution
  - si une seule activité répond aux critères, elle est automatiquement lancée
  - si plusieurs activités peuvent répondre, Android construit un chooser basé sur la valeur de l'Action, sur les Data et sur le Type.
  - si aucune app présente ne peut répondre, le système génère une exception : ActivityNotFoundException.

comment construire un Intent





comment construire un Intent

existence d'une app

- la vérification de la présente d'une app capable de répondre aux attentes peut se faire par un appel à la méthode resolveActivity

exemple

```
public void start(View view) {
    // IntentwebIntent = new Intent(Intent.ACTION_VIEW,Uri.parse(url));
    // startActivity(webIntent);

    Intent rzeczIntent = new Intent(ACTION_RZECZ,Uri.parse(url));
    PackageManager pm = getPackageManager();
    ComponentName thisComponentCanHandleMyRequest = rzeczIntent.resolveActivity(pm);
    if(thisComponentCanHandleMyRequest != null) startActivity(rzeczIntent);
    else {
        Toast.makeText(this, "No Activity Found, I change the action", Toast.LENGTH_LONG).show
        rzeczIntent.setAction(Intent.ACTION_VIEW);
        startActivity(rzeczIntent);
    }
}
```

récupération de résultats

récupération de résultats

du côté du lanceur

- une Activity lance une autre Activity (avec ou sans données) et souhaite récupérer un résultat, elle construit un Intent mais... au lieu de faire un appel à *startActivity(Intent)*, elle fait un appel à...

  startActivityForResult(Intent,CODE)

- le *CODE* permet de distinguer deux retours de résultats depuis deux Activity qui auraient pu être lancées depuis cette même Activity

récupération de résultats

du côté de la nouvelle Activity

- pour renvoyer ses résultats, la nouvelle Activity doit créer un Intent
  Intent ir = new Intent(), empaqueter le résultat
  ir.putExtra("myResult",resultat) et envoyer le résultat
  setResult(RESULT_OK, ir)
- *RESULT_OK* est le status du résultat il indique si tout s'est bien passé

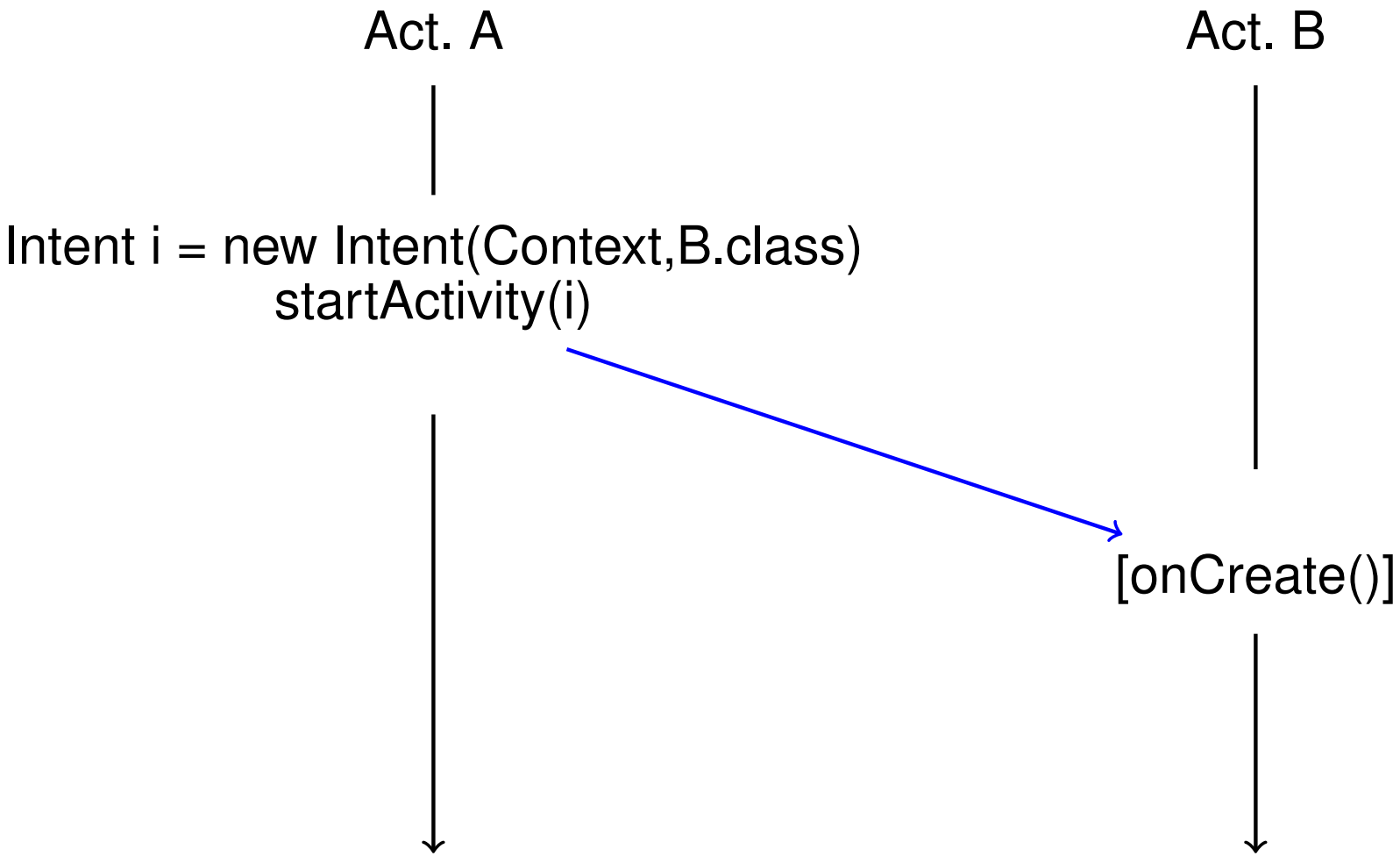
récupération de résultats

du côté du lanceur (de nouveau)

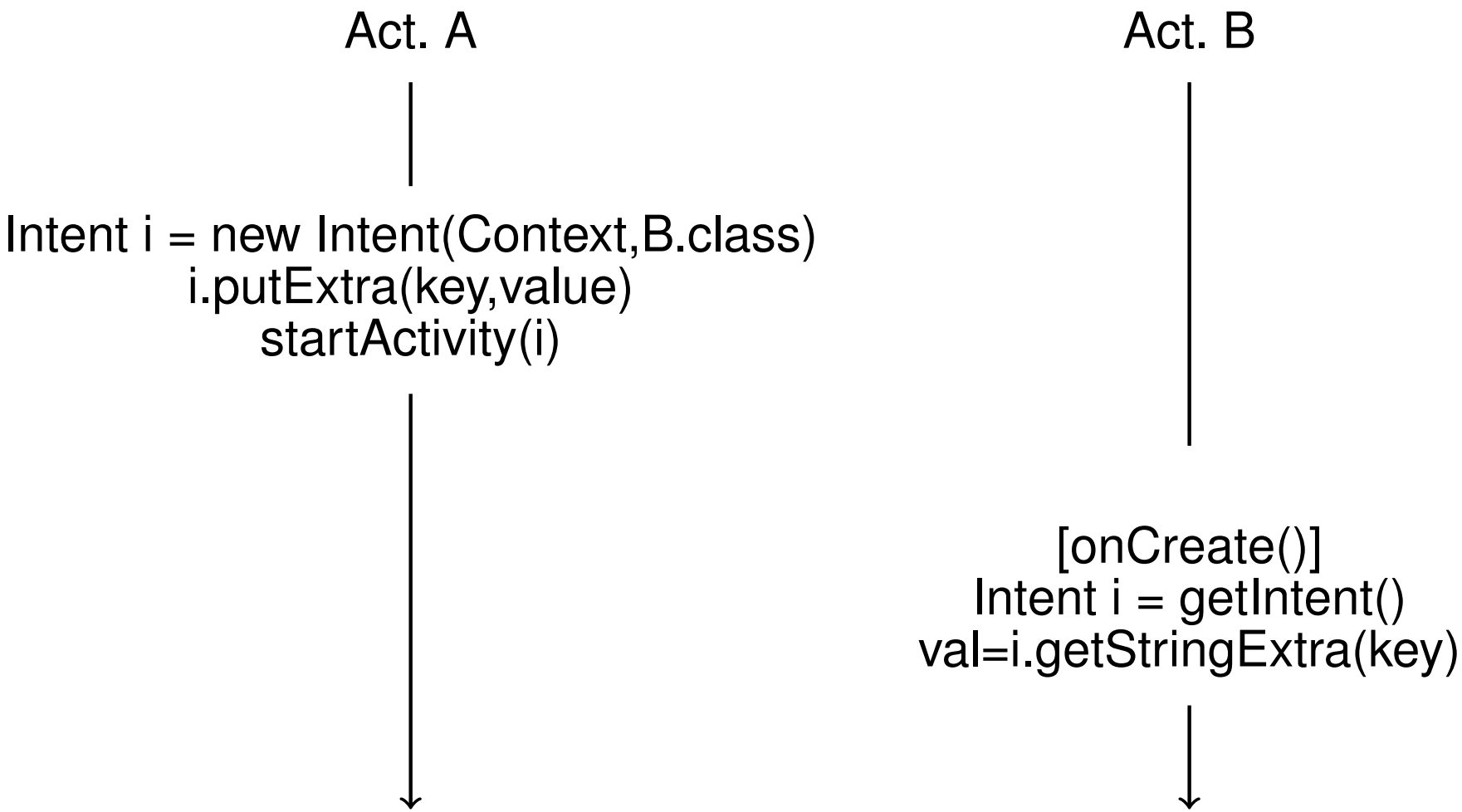
- pour récupérer le résultat, le lanceur doit implémenter la méthode *onActivityResult(int code,int returnCode,Intent resultat)*
- la récupération du résultat se fait avec un appel à la méthode *getIntExtra(clef,valeur-par-défaut)* ou *getStringExtra(...)*

en résumé

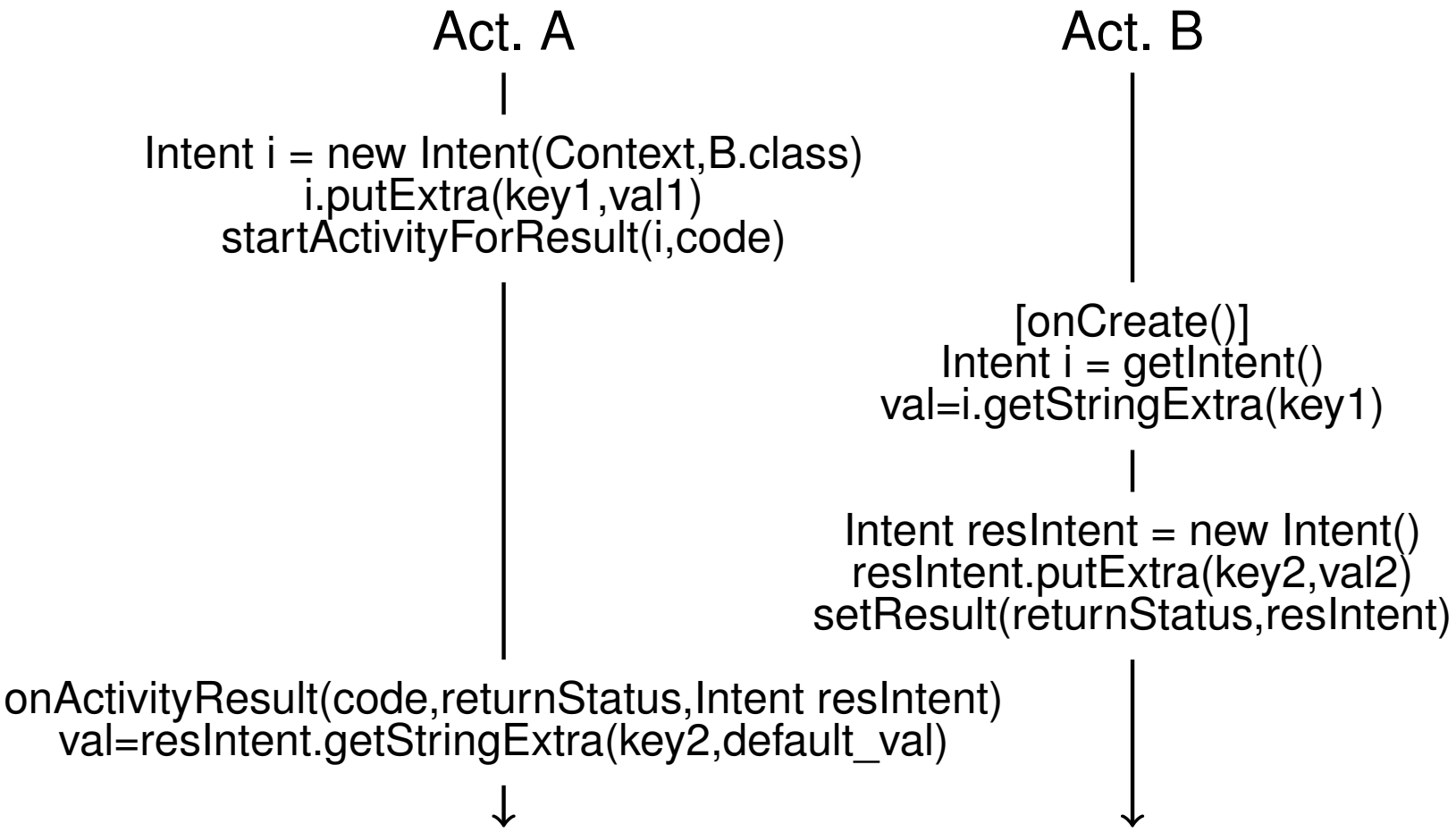
pour un Intent explicite sans données et sans retour



Intent explicite avec données et sans retour



Intent explicite avec données et résulats en retour



N'oubliez pas le fichier AndroidManifest.xml

lorsque dans votre projet vous avez plusieurs Activity, elles doivent toutes apparaitrent dans le fichier AndroidManifest.xml

```
<application
    android:allowBackup="true"
    android:icon="@drawable/ic_launcher"
    android:label="@string/app_name"
    android:theme="@style/AppTheme" >
    <activity
        android:name=".HomeActivity"
        android:label="@string/app_name" >
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
    <activity
        android:name=".ExplicitActivity"
        android:label="@string/title_activity_explicit" >
    </activity>
    <activity
        android:name=".CountActivity"
        android:label="@string/title_activity_count" >
    </activity>
</application>
```

Questions ?

Quelques références

- http ://developer.android.com/reference/android/content/Intent.html
- http ://developer.android.com/guide/components/intents-filters.html
- http ://developer.android.com/training/basics/firstapp/starting-activity.html
- http ://developer.android.com/training/basics/intents/sending.html