# אנדרואיד – שיעור 3

ישנם שני סוגים של כוונות (=intent):

Explicit- מפורש, מה שלמדנו עד כה, עוסק במעבר נתונים או מעבר בין דפים בתוך האפליקציה.
Implisit- מרומז, עוסק בהפעלה ושליטה בפונקציות של הטלפון מחוץ לאפליקציה.
למשל: חייגן, דפדפן, הגדרות GPS וכו'.
**אינטנט = כוונה.** (מכאן זה סיכום מהספר)
כוונה מרומזת אינה מפרטת רכיב ספציפי לשימוש, במקום זה היא מפרטת את הפונקציונליות הנדרשת דרך פילטר, והאנדרואיד מוכרח להחליט מהו הרכיב הטוב ביותר לשימוש.
פילטר האינטנט יכול להיות, פעולה, מידע או קטגוריה.
פילטר האינטנט הנפוץ ביותר לשימוש הוא הפעולה, והפעולה הנפוצה ביותר היא ACTION_VIEW
אופן בקשה זה מצריך משאב זהה המזהה URI ספציפי והצגת המידע למשתמש.
הוא עושה את הפעולה הסבירה ביותר עבור ה URI.
למשל: כשמתחיל ב- http או :tel הוא מסיק שעליו לפתוח דפדפן או חייגן.

**הצעדים להתחלת פעולה תוך שימוש ב- implicit intent:**

**1-** הצהרה על הכוונה עם פירוט הפילטר הדרוש

```
Intent intent  = new Intent (Intent.ACTION_VIEW);
```

- האנדרואיד מחלק את הפונקציונליות למשפחות ומציין זאת באמצעות מעין דגלים הכתובים בתוך הסוגריים הפנימיים של האינטנט- לכל מטרה יש את הפרמטרים שלה.

**2-** קישור כל מידע נוסף לבקשת האינטנט להרצת הפעולה.

```
intent.setData(Uri.parse("http://www.google.com"));
```

- יודע לטפל בנתיבים שקשורים לקבצים, action וכדומה, מאפשר parse לכל סוג שנרצה.

**3-** להעביר את האינטנט ל- startActivity(intent)

דוגמא נוספת:

```
Intent intent = new intent (Intent.ACTION_VIEW);
intent.setData(Uri.parse("tel:0548666556"));
stsrtActivity(intent);
```

**URI-** בד"כ סוג של נתיב, URI הוא סוג כללי שיודע לזהות ולפרש נתיבים. הוא פילטר ובין היתר גם מוודא את תקינות השורה.
**הרשאות**- לא כל ה-API דורשים הרשאות, כך שבמועד הורדת האפליקציה על המשתמש לאשר את ההרשאות על מנת להשתמש באפליקציה.
אם רוצים שקט ותמיכה בכל הדפדפנים אנו כן נוסיף הרשאה.

**הוספת הרשאה:**

בחירת הרשאה ספציפית -> uses permission -> add -> permissions -> Manifest

רישום ישיר ב- manifest:

```
<uses-permission android:name="andoid.permission.CALL_PHONE"/>
```

עבודת כיתה

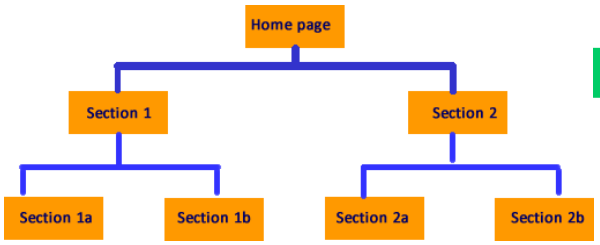
להוסיף את הבאים:
פתיחת אתר, חייגן, שליחת מייל, שיתוף, מצלמה והודעת טקסט.

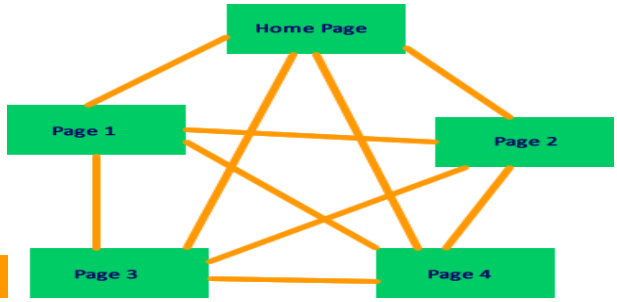
## העברת מידע בין אקטיביטיס

עד כה עבדנו עם startActivity, המשמעות של פונקציה זו מתבטאת באפליקציה בעלת מבנה עץ, המבנה זורם קדימה ואחורה- אך לא לצדדים והחזרה היא תמיד לנקודה ממנה הגענו (כפתור ה-back) – זוהי שיטה בינארית- ידוע מאין באתי ולאן אני הולך, שיטה שמטיבה עם הזיכרון ושומרת על המחסנית ריקה, ומקל על עבודתו של אוסף הזבל וחוסך בזמן ריצה.
השיטה השנייה היא שיטת ה- "מש" שבה יש מעבר מכל דף לכל דף, עד שלעיים קרובות יש בלאגן והמתכנת עלול ללכת לאיבוד ולהתבלבל, הוא לא בטוח איזה דף שולח לאן. יתרה מכך, המחסנית מתנפחת ככל שהמעבר תכוף יותר בין הדפים, אנדרואיד מזמן את אוסף הזבל לעיתים קרובות וכל זה מתבטא בזמן הריצה של האפליקציה וגורע מחווית המשתמש.

אז ראינו שני מבנים: מבנה עץ ומבנה מש, לכל אחד יתרונות וחסרונות והנושא הנוכחי עוסק במצב שבו נדרש מעבר בין דפי האפליקציה בשיטת המש, שבו דף אחד מפעיל דף אחר בציפייה לקבל ממנו תוצאה כלשהיא.
בשביל הפשטות נקרא לדף הראשון "המתקשר" ולדף השני "הפועל".
מה שייחודי ומפליא בשיטה שלפנינו הוא שהפועל מושמד עם סיום פעולתו ולכן אין עומס על אוסף הזבל. זה קצת מסובך הסיפור הזה אז אני יציג תרשים כללי ואז נצלול לעומק התחביר:

המתקשר מתחיל פעולה שמצפה לאיזו שהיא תוצאה---> מעמיס מידע באינטנט ---> פקודה – צא לדרך ותביא תוצאות!
הפועל מקבל את השיחה ----> קולט את המידע שבאינטנט --> ממיר את המידע ----> שולח את התוצאה ---> משמיד את עצמו.




האיורים ממחישים את ההבדל בין מבנה העץ ובין מבנה המש.
מבנה המש מורכב מאוד וטומן סכנה גדולה של תוצאות לא רצויות- וכאן באים לעזרתנו הדגלים שמוודאים איזה אקטיביטי הופעל (פועל) ולהיכן חזרו התוצאות (מתקשר).

יש מספר דרכים להעברת מידע בין אקטיביטיס:
א- הצהרה על משתנים באקטיביטי שמתחיל ולשלוח לאקטיביטי השני.
ב- להעמיס תוספת על האינטנט המשמש להתחלת אקטיביטי תוך שימוש בחבילה.
ג- שימוש במאגר נתונים (=נלמד בהמשך הקורס).

**<u>התחביר :</u>** (יש דוגמא בספר, עמודים 166-167)

<u>בקובץ **MainActivity** (המתקשר) נרשום את הפונקציות הבאות:</u>

**public void startActivityForResultClick(View target){**

פונקציה שמתחילה את האקטיביטי הממתינה לתוצאות

**Private static final int A_B = 1;**

זה דגל ראשון המעיד על כך שהמידע נשלח לכתובת הנכונה – ניתן לו איזה שם שנרצה.

**Intent intent = new Intent (this, Second.class);**

הגדרת אינטנט שיפעיל את האקטיביטי השני (הפועל)

**srartActivityForResult(intent, A_B);**

ציווי – התחל פעולה.

**}**

**public void onActivityResult (int requestCode, int resultCode, Intent data) {**

פונקציה שמקבלת חזרה את המידע ואת הדגלים

**if (requestCode == A_B && resultCode == RESULT_OK) {**

אל הלולאה מתקבלים הדגלים וכאן נבדק שאכן התקבלו הדגלים הנכונים.
בתוך הסוגריים המסולסלים נזין את ערכי המשתנים שהתקבלו, או נפרוק את המידע מהאינטנט שחזר אלינו.

**}**

**super.onActivityResult (requestCode, resultCode, data);**

קונסטרקטור עצמי – אני לא בטוחה מה עושה ולמה הוא כאן.

**}**

<u>בקובץ **Second** (הפועל) נרשום את הפונקציות הבאות:</u>

בתוך פונקציית onCreate, אחרי כתיבת super ואחרי setContentView נוסיף:

**Intent intent = getIntent ();**

לשים לב שאין כאן new! אלא זה מקום פריקת ה-intent מהמתקשר.
בשורות הבאות נפרוק את המידע ונסגור את פונקציית onCreate

**finish();**

הוראה לקובץ הפועל לסגור את עצמו.

## Dialogs

הדיאלוג הוא חלון קטן שקופץ בקדמת המסך, שכבה מעל הפעילות הנוכחית.
חלון הדיאלוג נקבל את כל האינטרקציה עם המשתמש ומאפיל על ה-MainActivity הפעיל.
בדרך כלל יופיע חלון הדיאלוג כדי לבחון עניין ספציפי מול המשתמש הקשור ישירות לאפליקציה שרצה.
הדיאלוג הוא מחלקה בסיסית ליצירת דיאלוגים, אך לא אופייני להשתמש ישירות, סביר יותר להשתמש באחת מהמחלקות היורשות של מחלקת הדיאלוג, לא לשכוח להוסיף למחלקות הפנימיות extens ובתוכו- setContentView, המחלקות היורשות:

**AlertDialog**- דיאלוג שיכול לנהל בין 0 ל- 3 כפתורים ו / או רשימה של פריטים לבחירה שעשויים לכלול כתורי בחירה או כפתורי רדיו.
מחלקה זו נותנת את המבנה הנפוץ ביותר לממשק המשתמש.

```
AlertDialog.Builder dialog = new AlertDialog.Builder(MainActivity.this);
Dialog.setTitel("alert!");
Dialog.setMessage("ffdvv");
Dialog.setPositiveButton("ok",null);
Dialog.setNeutralButton("ok",null);
alog.setPositiveButton("no",null);
dialog.show();
```

**ProgressDialog**- מציג גלגל התקדמות או בר התקדמות. מאחר והוא תולדה של alertDialog הוא תומך גם כן בכפתורים.
**DataPickerDialog**- מאפשר למשתמש לבחור תאריך.
**TimePickerDialog**- מאפשר למשתמש לבחור שעה.

### השיטות המשמשות לדיאלוג:

create() – יצירת ה – AlertDialog באמצעות שיטה זו במחלקה הפנימית שלו: AlertDialog.Builder
setMessage()- משמשת להגדרת הטקסט הספציפי שיופיע בדיאלוג.
setTitle() & setIcon()- שיטות אלו משמשות להגדרת הכותרת או האייקון שבדיאלוג.
show()- השיטה שגורמת להקפצת הדיאלוג על המסך.
setButton()- הוספת כפתורים לתיבת הדיאלוג.
onClick()- על מנת להגדיר את הפעולה שיעשה הכפתור הממוקם בתיבת הדיאלוג נשתמש בפונקציית הקולבק כדי להוציא לפועל את ה- DialogInterface.OnClickListener class
מימוש הכפתורים: ע"י ליסטנר:

```
Dialog.setNeutralButton("ok",new DialogInterface.OnClickListener());
```

פונקציה שמתעוררת כאשר לוחצים על כפתור back:

```
Public void onBackPressed(){
openAlertDialog();
}
```

### מחלקה פנימית – דיאלוג.

נבנה מחלקה פנימית של דיאלוג, נוסיף exteneds לדיאלוג
בתוך המחלקה הפנימית: ב- onClick נצהיר על אובייקט חדש דיאלוג: mainActivity.this
כעת שהוא קיים נוכל לקרוא ל- ()show – והוא יפתח את הדף שעיצבנו בתיבת אלרט.