Xamarin Evolve 2014

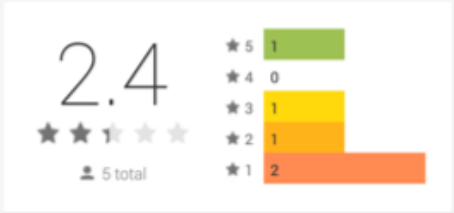
Android Performance

Rob Gibbens
rob.gibbens@xamarin.com
@RobGibbens

Xamarin University

Why performance matters

- Provide a solid experience for your app
- Users expect the app to function at the same performance levels as other apps
- Bad performance will equal bad reviews

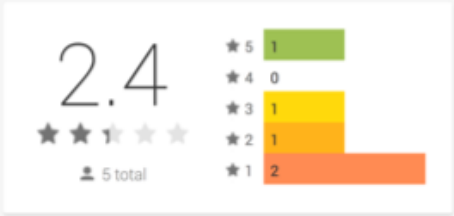


Xamarin University

Why performance matters

- Provide a solid experience for your app
- Users expect the app to function at the same performance levels as other apps
- Bad performance will equal bad reviews

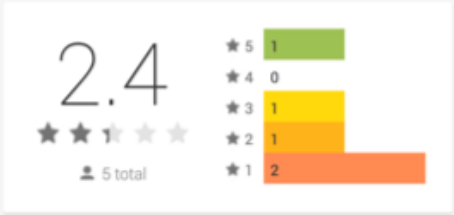
Xamarin University



Why performance matters

- Provide a solid experience for your app
- Users expect the app to function at the same performance levels as other apps
- Bad performance will equal bad reviews

Xamarin University



Agenda

1. Standard Tips
2. Async all the things!
3. Garbage Collection
4. ListView optimizations
5. ViewHolder pattern

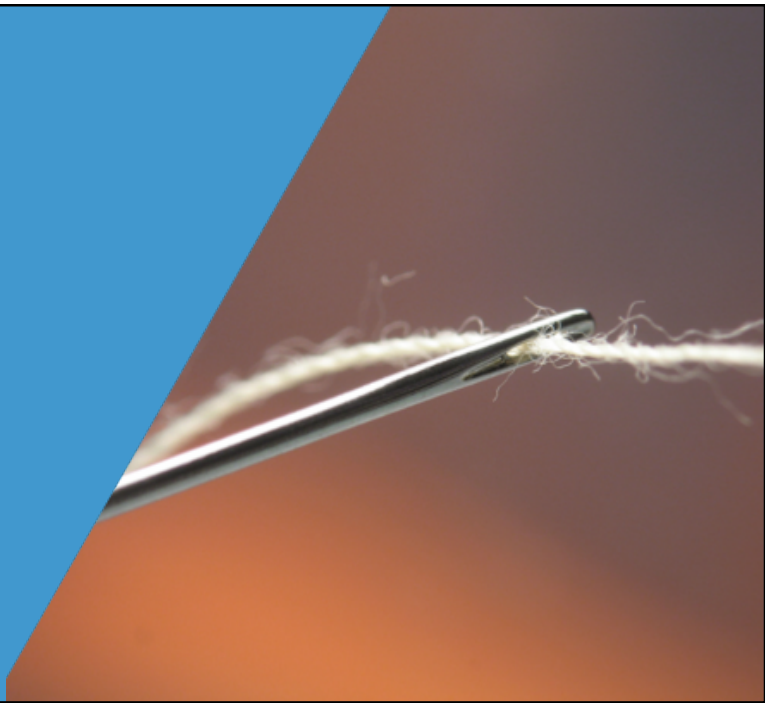
Xamarin University

Standard Tips

Xamarin University

Standard Tips

- Don't abuse the UI thread



Xamarin University

Standard Tips

- Don't abuse the UI thread
- **Think about the garbage collector**

Xamarin University

Standard Tips

- Don't abuse the UI thread
- **Think about the garbage collector**

Xamarin University

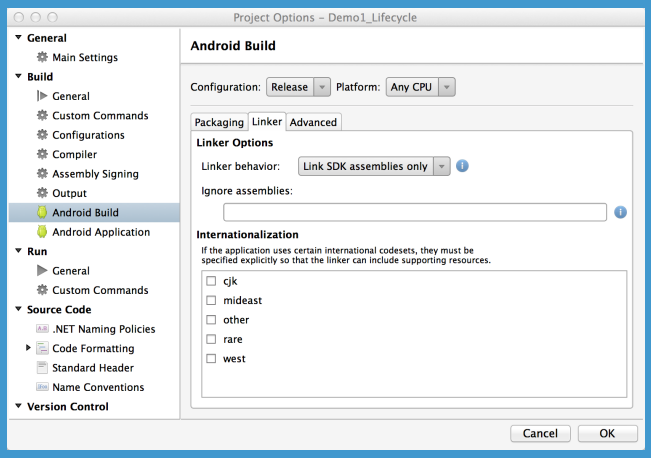
Standard Tips

- Don't abuse the UI thread
- **Think about the garbage collector**

Xamarin University

Standard Tips

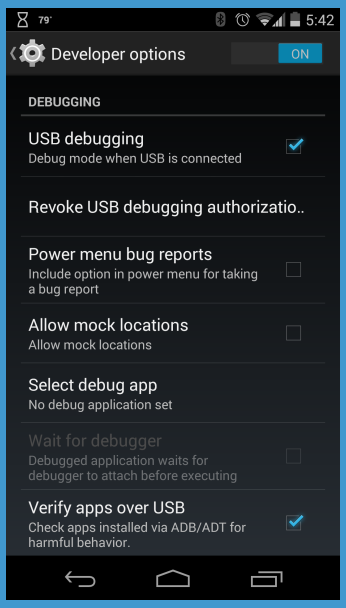
- Don't abuse the UI thread
- Think about the garbage collector
- **Use the right build settings**



Xamarin University

Standard Tips

- Don't abuse the UI thread
- Think about the garbage collector
- Use the right build settings
- **Utilize the Developer Options**



Xamarin University

Standard Tips

- Don't abuse the UI thread
- Think about the garbage collector
- Use the right build settings
- Utilize the Developer Options
- **Always test on physical devices**

Xamarin University

Standard Tips

- Don't abuse the UI thread
- Think about the garbage collector
- Use the right build settings
- Utilize the Developer Options
- Always test on physical devices
- **Test with large datasets**

Xamarin University

Agenda

1. ~~Standard Tips~~
2. **Async all the things!**
3. Garbage Collection
4. ListView optimizations
5. ViewHolder pattern

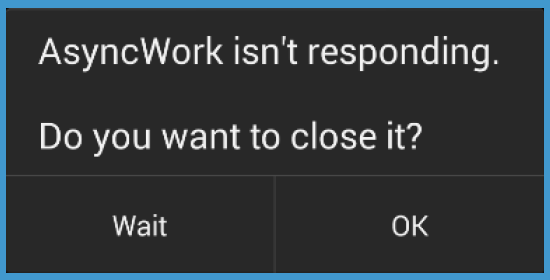
Xamarin University

async/await

Xamarin University

async/await

- If UI thread is working on your task, it isn't working for the UI
- Return control to the UI thread as soon as possible



Xamarin University

async/await

```
public async static Task<List<Song>> GetSongsAsync ()
{
}
```

Xamarin University

async/await

```
public async static Task<List<Song>> GetSongsAsync ()
{
    var client = new WebClient ();

    var data = await client.DownloadStringTaskAsync (bigFile);



}
```

Xamarin University

async/await

```
public async static Task<List<Song>> GetSongsAsync ()
{
    var client = new WebClient ();

    var data = await client.DownloadStringTaskAsync (bigFile);

    return JsonConvert.DeserializeObject<List<Song>> (data);

}
```

Xamarin University

async/await

```
public async static Task<List<Song>> GetSongsAsync ()
{
    var client = new WebClient ();

    var data = await client.DownloadStringTaskAsync (bigFile);

    return await Task.Run (() =>
        JsonConvert.DeserializeObject<List<Song>> (data)
    );
}
```

Xamarin University

Demo

Xamarin University

Agenda

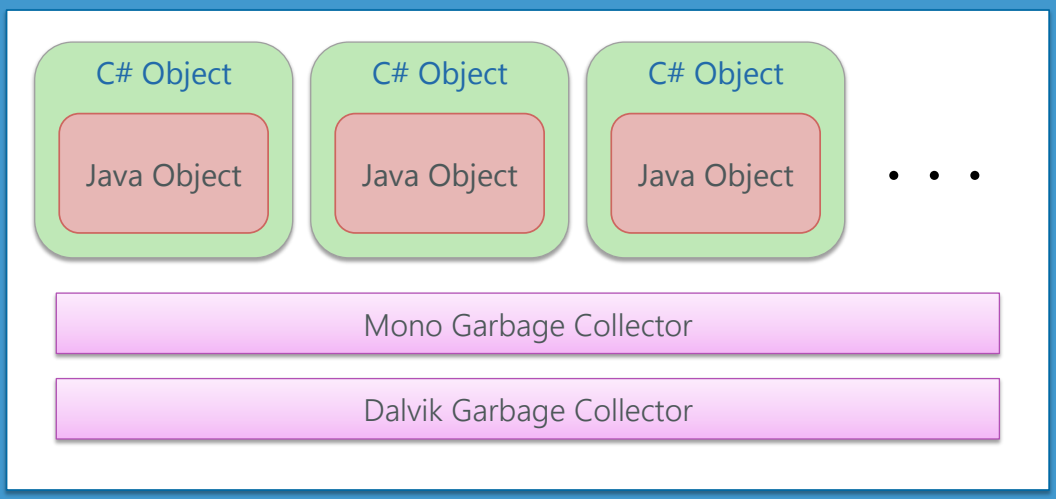
1. ~~Standard Tips~~
2. ~~Async all the things!~~
3. **Garbage Collection**
4. ListView optimizations
5. ViewHolder pattern

Xamarin University

Memory and Garbage Collection

Xamarin University

Architecture



Xamarin University

Garbage Collection

!=

Xamarin University

Garbage Collection

- Garbage Collection is used to manage your memory in your application
- Blocks the UI thread
- Mapped on top of the platform memory management

Xamarin University

Be GC Friendly

- Dispose objects that wrap scare resources
- Beware of event handlers - the can keep subscribed objects alive
- Beware of immortal objects due to strong reference cycles
- Dispose objects that hold large native things

Xamarin University

Be GC Friendly

- Dispose objects that wrap scare resources
- Beware of event handlers - the can keep subscribed objects alive
- Beware of immortal objects due to strong reference cycles
- Dispose objects that hold large native things

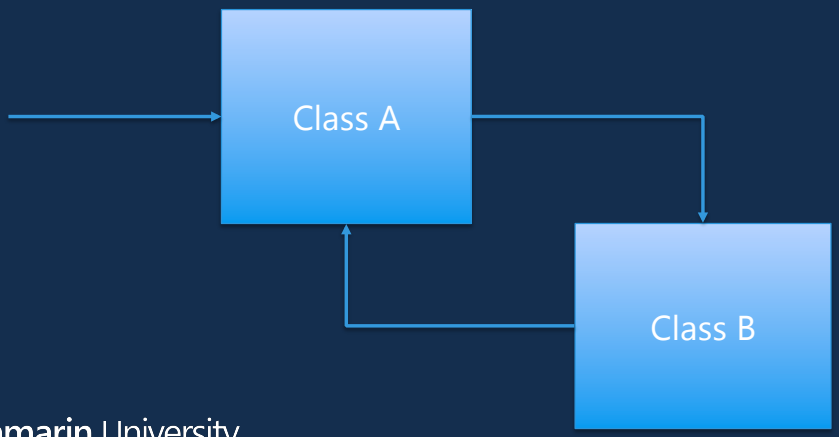
Xamarin University

Be GC Friendly

- Dispose objects that wrap scare resources
- Beware of event handlers - the can keep subscribed objects alive
- Beware of immortal objects due to strong reference cycles
- Dispose objects that hold large native things

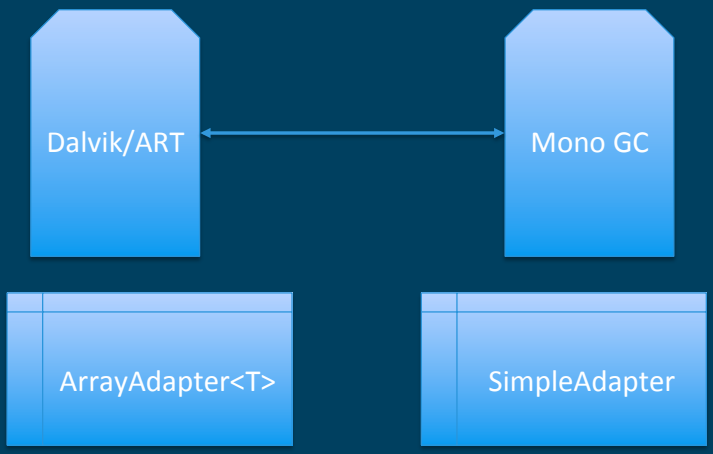
Xamarin University

GC Issue – Circular References



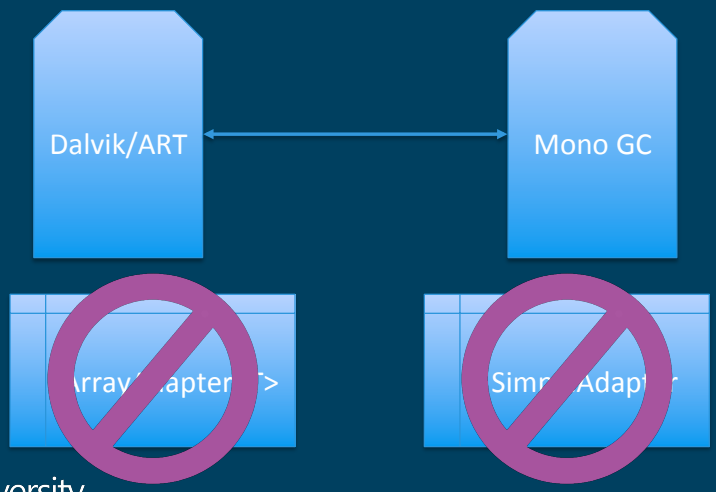
Xamarin University

GC Issue – Excessive Communication



Xamarin University

GC Issue – Excessive Communication



Xamarin University

GC Issue – Native Resources

Native Resources should be released asap

```
imageView.SetImageResource (Resource.Drawable.bigImage);
```

Internal Memory: 80 bytes

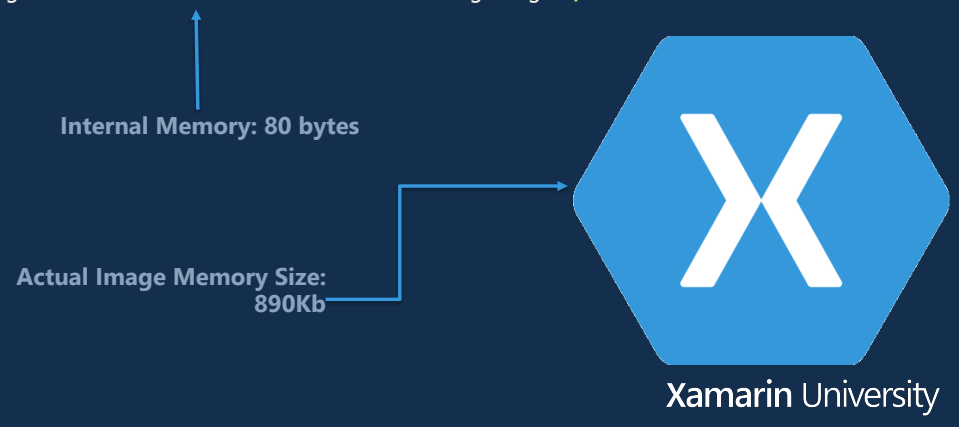
Actual Image Memory Size: 890Kb

Xamarin University

GC Issue – Native Resources

Native Resources should be released asap

```
imageView.SetImageResource (Resource.Drawable.bigImage);
```



TODO

```
class ViewHolder : Java.Lang.Object {
    public TextView FirstNameView;
    public TextView LastNameView;
    public ImageView PortraitView;
}
```

Xamarin University

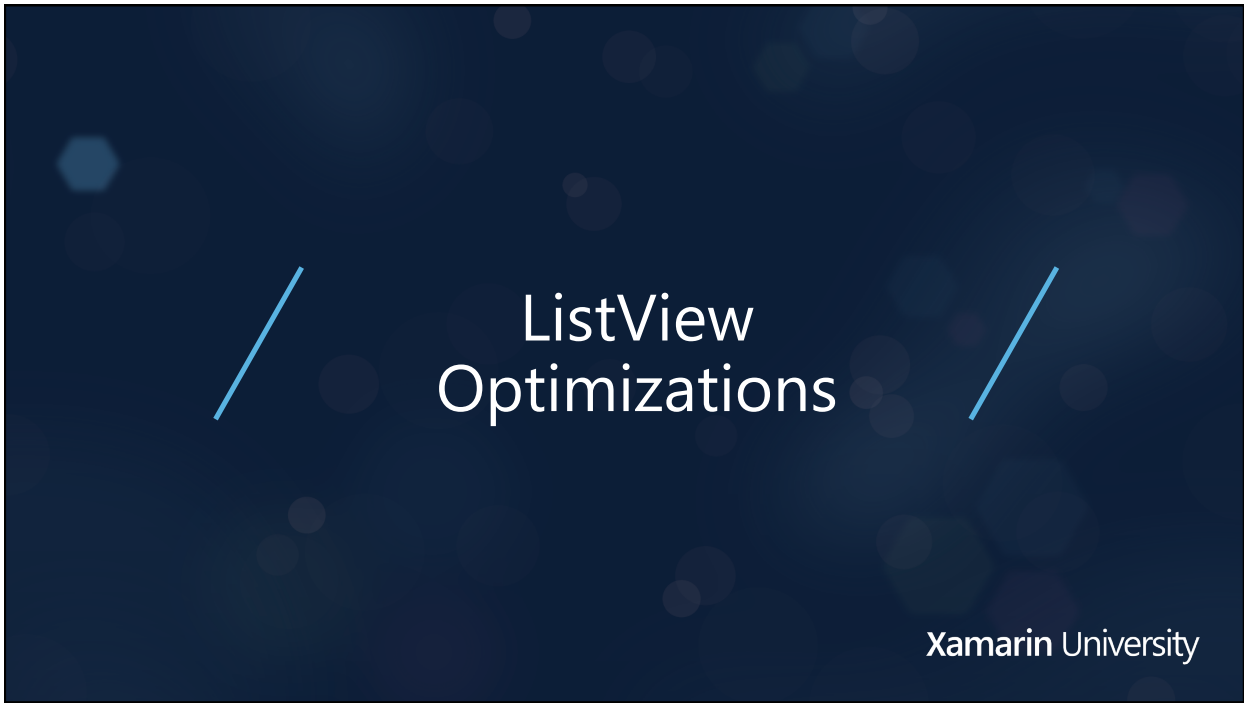
Demo

Xamarin University

Agenda

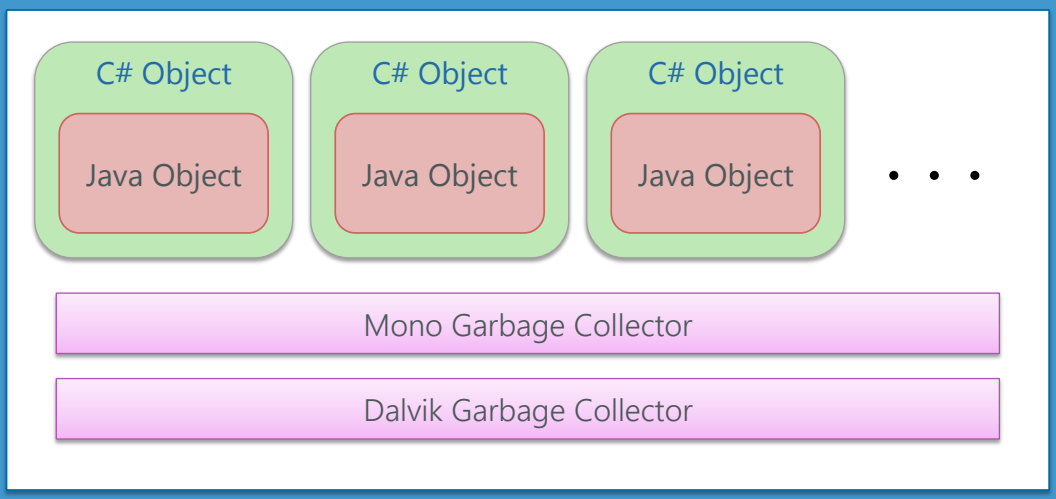
1. ~~Standard Tips~~
2. ~~Async all the things!~~
3. ~~Garbage Collection~~
4. **ListView optimizations**
5. ViewHolder pattern

Xamarin University



ListView
Optimizations
Xamarin University

ArrayAdapter



Xamarin University

Custom Adapters

```
public class CustomAdapter : BaseAdapter<Session>
{
    public CustomAdapter (Context context, IEnumerable<Session> data)
    {
    }
}
```

Xamarin University

Custom Adapters

```
public override View GetView (int position, View convertView,
                                                    ViewGroup parent)
{




}
```

Xamarin University

Custom Adapters

```
public override View GetView (int position, View convertView,
                                                    ViewGroup parent)
{
    View view = convertView;
    if (view == null) {
        //Create new View
    }


}
```

Xamarin University

Custom Adapters

```
public override View GetView (int position, View convertView,
                                                    ViewGroup parent)
{
    View view = convertView;
    if (view == null) {
        //Create new View
    }
    //Lookup data for row
    //Use FindViewById<T> to find UI elements
    //Populate views with data
}
```

Xamarin University

Demo

Xamarin University

Agenda

1. ~~Standard Tips~~
2. ~~Async all the things!~~
3. ~~Garbage Collection~~
4. ~~ListView optimizations~~
5. **ViewHolder pattern**

Xamarin University

View Holder Pattern

Xamarin University

Custom Adapters

```
public override View GetView (int position, View convertView,
                                                ViewGroup parent)
{
    View view = convertView;
    if (view == null) {
        //Create new View
    }
    //Lookup data for row
    //Use FindViewById<T> to find UI elements
    //Populate views with data
}
```

Xamarin University

View Holder Pattern

```
class ViewHolder : Java.Lang.Object {
    public TextView FirstNameView;
    public TextView LastNameView;
    public ImageView PortraitView;
}
```

Xamarin University

View Holder Pattern

```
if (view == null) {
    view = LayoutInflater.Inflate (layout);

    viewHolder = new ViewHolder ();

    viewHolder.FirstNameView
            = view.FindViewById<TextView> (Resource.Id.title);

    /* Cache other references */
    view.Tag = viewHolder;
} else {
    viewHolder = view.Tag as ViewHolder;
}
```

Xamarin University

View Holder Pattern

```
if (view == null) {
    viewHolder = new ViewHolder ();
} else {
    viewHolder = view.Tag as ViewHolder;
}

var person = data [position];

viewHolder.FirstNameView.Text = person.FirstName;
viewHolder.LastNameView.Text = person.LastName;
```

Xamarin University

Demo

Xamarin University

Summary

1. Standard Tips
2. Async all the things!
3. Garbage Collection
4. ListView optimizations
5. ViewHolder pattern

Xamarin University

Related Sessions

- Applying C# Async in Mobile
  Wednesday, 1:15 - Joe Mayo
- Scrolling Performance and Bitmap Caching in Android
  Wednesday, 5:00 - Brett Duncavage
- Memory Management Tips & Tricks for iOS
  Thursday, 3:15 - Rodrigo Kumpera

Xamarin University

Questions?

Xamarin University

Xamarin Evolve 2014

Android Performance

Rob Gibbens
rob.gibbens@xamarin.com
@RobGibbens

Xamarin University