KANDULA SRINIVASA REDDY MEMORIAL COLLEGE OF ENGINEERING
(AUTONOMOUS)

KADAPA-516003. AP

(Approved by AICTE, Affiliated to JNTUA, Ananthapuramu, Accredited by NAAC)

(An ISO 9001-2008 Certified Institution)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATION COURSE
ON
"ANDROID APPLICATION DEVELOPMENT"

Resource Person : **1. Mr. G. Nagendra Babu, Assistant Professor, Dept. of CSE, KSRMCE**

2. Mr. S. Kaja Khizar, Assistant Professor, Dept. of CSE, KSRMCE

Course Coordinator: Mrs. S. Riyaz Bhanu, Assistant Professor, Dept. of CSE, KSRMCE

Duration: **09/09/2019 to 27/09/2019**

K.S.R.M. COLLEGE OF ENGINEERING

(UGC - AUTONOMOUS)

Kadapa, Andhra Pradesh, India - 516003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution

Lr./KSRMCE/ (Department of CSE)/2019-20

Date: 03/09/2019

To
The Principal
KSRM College of Engineering
Kadapa, AP.

Sub: KSRMCE - (Department of CSE) – Permission to conduct certification course on Android Application Development - Requested – reg.

---***---

Respected Sir,

With reference to the cited, the Department of CSE is planning to conduct certificate course on "**Android Application Development**" for B.Tech students from 09/09/2019 to 27/09/2019. So I request you to grant permission to conduct the certificate course. This is submitted for your kind perusal.

Thanking you sir,

Yours Faithfully,
Riyaz Bhanu
Coordinator,
Smt. S. Riyaz Bhanu,
Assistant Professor ,
CSE Dept.,

Forwarded to the Principal Sir,

Permitted
U.S.S.Murty
03/09/2019

Cc:

To The Director for Information

To All Deans/HODs

K.S.R.M. COLLEGE OF ENGINEERING

(UGC - AUTONOMOUS)

Kadapa, Andhra Pradesh, India - 516003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution

Dated: 04/09/2019

All the B.Tech students are here by informed that department of Computer Science & Engineering is going to organize certification course on **"Android Application Development"** from 09/09/2019 to 27/09/2019. Interested students do register their names with the below mentioned coordinator on or before 07/09/2019, 5PM.

For any queries contact,

Coordinator:
Smt. S. Riyaz Bhanu,
Assistant Professor,
CSE Dept.,

HOD

Dr. M. Sreenivasulu,
M. E., Ph. D.
Professer & HOD CSE
K.S R.M. College of Engineering
K A D A P A - 516 003

Cc to:

The Management /Director / All Deans / All HODS/Staff / Students for information

The IQAC Cell for Documentation

K.S.R.M. COLLEGE OF ENGINEERING

(UGC - AUTONOMOUS)

Kadapa, Andhra Pradesh,India - 516003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution

Date:04.09.2019

Department of Computer Science & Engineering

Certificate Course on Android Application Development (09/09/2019 to 27/09/2019)

Registered Student List

S.No.	Roll Number	Name of the Student	Year & Branch	Signature

Coordinator **HOD**

K.S.R.M. COLLEGE OF ENGINEERING

(UGC - AUTONOMOUS)

Kadapa, Andhra Pradesh,India - 516003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution

Date:04.09.2019

Department of Computer Science & Engineering

Certificate Course on Android Application Development (09/09/2019 to 27/09/2019)

Registered Student List

S.No.	Roll Number	Name of the Student	Year & Branch	Signature
1	169Y1A0503	Abburi vanitha	B. Tech VII Sem	
2	169Y1A0505	Alamuru Shireesha	B-Tech VII Sem	
3	169Y1A0507	A Subrahmanya Sumanth Kumar	B-Tech VII Sem	
4	169Y1A0505	A Venkata Ravindra Reddy	B. tech VII sem	
5	169Y1A0523	Chukka swapna	B-Tech VII sem	Chukka swapna
6	179Y1A0542	Drasham Naveen	B.Tech V sem	
7	169Y1A0510	B. Govardhan Reddy	7th sem	B. Govardhan
8	179Y1A0546	G. MAMATHA	B.Tech. V sem	
9	179Y1A0541	Donthu sukanya	B-Tech V sem	
10	179Y1A0544	Abhishek kumar shaw	B Tech V sem	
11	169Y1A0502	A. Sreelanth	B.Tech VII sem	
12	179Y1A0543	D. Jyoshna	B Tech V sem	
13	169Y1A0511	B. ANAND SRINIVASA YADAV	7th sem	
14	179Y1A0545	E. poudri.	B Tech V sem	poudri
15	169Y1A0524	Dalaie vasudha	B. Tech 7th sem	
16	169Y1A0509	B. Venkatesh	VII sem	

17	179Y1A0570	Kuruba Keerthi	B.Tech Vsem	K. Keerithi
18	169Y1A0578	pamidi Gowtham Reddy	B.tech 4th sem	Gow
19	179Y1A0564	Kancherla Sasikala	B.Tech Vsem	K. Sasikala
20	179Y1A0522	C. Lohitha kumar	B.Tech Vsem	Lohitha
21	179Y1A0548	G. Haritha	B.Tech Vsem	Hari
22	179Y1A0568	Kotluru Noor Basha	B.Tech Vsem	K. Noor Basha
23	179Y1A0521	cheenu Brahmani	B.Tech Vsem	Brah
24	179Y1A0537	D. Yogesh	B.Tech V sem	Yogi
25	169Y1A0579	pappireddy Vijitha	B.tech 4th	Vijitha
26	179Y1A0523	chanda Harinadh Reddy	B.Tech Vsem	Harinadh
27	179Y1A0569	Kuntavala prithvi	B.Tech Vsem	K. Prithvi
28	179Y1A0538	D. Anusha.	B.Tech V sem	Anu
29	179Y1A0549	G. VINOD KUMAR.	B.Tech V sem	Vinod
30	179Y1A0524	C. harsha vardhan reddy	B.Tech Vsem	Harshavarthan
31	179Y1A0567	K. Siva Narayana	B.Tech Vsem	K. Siva Narayana
32	179Y1A0547	G. Sudesh chaudhary	B.Tech Vsem	Sudesh
33	179Y1A0539	Doddi Devananda	B.Tech V sem	Dev
34	179Y1A0525	chawa Sravani	B.Tech Vsem	Sravani
35	179Y1A0566	Kancharveela Harika	B.Tech Vsem	K. Harika
36	179Y1A0529	chilakala Mahamuadrafi	B.Tech Vsem	Muu
37	179Y1A0540	Yasaswini	B.Tech V sem	Yas
38	179Y1A0565	K. Venkata Subba Reddy	B.Tech Vsem	K. Subba Reddy
39	179Y1A0528	C. Shaik Mahammad Aslam	B.Tech Vsem	Moula
40	179Y1A0550	C. Maheswari	B.Tech V sem	Maheswari

41	169Y1A0549	Ketham vara lakshmi	B.tech 7th sem	Ketham
42	179Y1A0532	DMohammad Shoaib	B.Tech V sem	
43	179Y1A0559	K. Ramacharan	B.Tech V sem	Ramacharan
44	179YA0557	J. Haritha	B.Tech V sem	Haritha
45	179YA0554	P. Harini	B.Tech V sem	Harini
46	169Y1A0576	Palempalli veeraprasana	7th	Prasana
47	179YA0556	I Ram Bhupal Reddy	B.Tech V sem	Ram Bhupal Reddy
48	179YA0538	J. Sai kumar	B.Tech V sem	Sai kumar
49	179Y1A0536	Dandari venkateshwarlu	B.Tech V sem	Venkat
50	179Y1A0521	B Ramana Sai	B.Tech V sem	Ramana
51	179Y1A0533	Dabbesa Soumya	B.Tech V sem	Sow
52	179Y1A0552	G. Rajavardhan Reddy	B.Tech V sem	Rajavardhan
53	169Y1A0550	Kokatam maheswar eddy	B.Tech 7th sem	Kokatam
54	179Y1A0535	Dandu sai Kiran	B.Tech V sem	Kiran
55	169YA0577	Palleti venkata yeseshwini	B.Tech 7th sem	Yeseshwini
56	179Y1A0555	I. Manoj kumar	B.Tech 7th sem	Manoj
57	179Y1A0553	G. Sai Sujitha Reddy	B.Tech V sem	Sai Sujitha
58	179Y1A0531	chinthala Akhila	B.Tech V sem	Akhila
59	169Y1A0551	Kondaru Babitha	B.Tech VII sem	
60	179Y1A0549	G. Vinod kumar	B.Tech V sem	Vinod
61	179Y1A0557	G. Ajay Kumar	B.Tech V sem	Ajay kumar
62	179Y1A0534	Daddanala Yogya vignesh	B.Tech. V sem	Yogya
63	179Y1A0528	Bosigari Bharathi	B.Tech V sem	Bharathi
64	179Y1A0550	G. Sai Kiran Reddy	B.Tech V sem	Sai kiran

65	169YIA0514	Boggarapu Nagamanasa	B-Tech VIIsem	B
66	179YIA5853	Kamineni keerthana	B-Tech Vsem	Keerthana
67	179YIA561	K. Vasanthi Reddy	B-Tech Vsem	Vasanthi
68	179YIA560	K. kavitha Reddy	B-Tech Vsem	kavitha
69	179YIA565	K.venkata Sai kumar reddy	B-tech Vsem	Sai kumar Reddy
70	169YIA0515	Bojja Bhagyasree	B-Tech VIIsem	Bhagya

Rajasekhar
Coordinator

HOD

Dr. M. Sreenivasulu,
M. E., Ph. D.
Professor & HOD CSE
K.S.R.M. College of Engineering
K A D A P A - 516 003

K.S.R.M. COLLEGE OF ENGINEERING

(UGC - AUTONOMOUS)

Kadapa, Andhra Pradesh,India - 516003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution

Department of Computer Science & Engineering
Certificate Course on Android Application Development
Registered Student List

S.No.	Roll Number	Name of the Student	Year & Branch	Email id
1	169Y1A0502	Aakuleti Sreekanth	B.Tech VII SEM	169Y1A0502@ksrmce.ac.in
2	169Y1A0503	Abburi Vanitha	B.Tech VII SEM	169Y1A0503@ksrmce.ac.in
3	169Y1A0504	Alamuru Shireesha	B.Tech VII SEM	169Y1A0504@ksrmce.ac.in
4	169Y1A0505	AVenkata Ravindra Reddy	B.Tech VII SEM	169Y1A0505@ksrmce.ac.in
5	169Y1A0507	A Subrahmanya Sumanth Kumar	B.Tech VII SEM	169Y1A0507@ksrmce.ac.in
6	169Y1A0509	Bandla Venkatesh	B.Tech VII SEM	169Y1A0509@ksrmce.ac.in
7	169Y1A0510	B Govardhan Reddy	B.Tech VII SEM	169Y1A0510@ksrmce.ac.in
8	169Y1A0511	B Anandsrinivasyadav	B.Tech VII SEM	169Y1A0511@ksrmce.ac.in
9	169Y1A0514	Boggarapu Nagamanasa	B.Tech VII SEM	169Y1A0514@ksrmce.ac.in
10	169Y1A0515	Bojja Bhagya Sree	B.Tech VII SEM	169Y1A0515@ksrmce.ac.in
11	169Y1A0523	Chukka Swapna	B.Tech VII SEM	169Y1A0523@ksrmce.ac.in
12	169Y1A0524	Dalaie Vasudha	B.Tech VII SEM	169Y1A0524@ksrmce.ac.in
13	169Y1A0549	Ketham Vara Lakshmi	B.Tech VII SEM	169Y1A0549@ksrmce.ac.in
14	169Y1A0550	Kokatam Maheswarreddy	B.Tech VII SEM	169Y1A0550@ksrmce.ac.in
15	169Y1A0551	Konduru Babitha	B.Tech VII SEM	169Y1A0551@ksrmce.ac.in
16	169Y1A0576	Palempalli Veeraprasanna	B.Tech VII SEM	169Y1A0576@ksrmce.ac.in
17	169Y1A0577	Palleti Venkata Yeseshwini	B.Tech VII SEM	169Y1A0577@ksrmce.ac.in
18	169Y1A0578	Pamidi Gowtham Reddy	B.Tech VII SEM	169Y1A0578@ksrmce.ac.in
19	169Y1A0579	Pappireddy Vijitha	B.Tech VII SEM	169Y1A0579@ksrmce.ac.in
20	179Y1A0520	Bosigari Bharathi	B.Tech V SEM	179Y1A0520@ksrmce.ac.in
21	179Y1A0521	B Ramana Sai	B.Tech V SEM	179Y1A0521@ksrmce.ac.in
22	179Y1A0522	C.Lohith Kumar	B.Tech V SEM	179Y1A0522@ksrmce.ac.in
23	179Y1A0523	Chanda Harinadh Reddy	B.Tech V SEM	179Y1A0523@ksrmce.ac.in
24	179Y1A0524	C Harsha Vardhan Reddy	B.Tech V SEM	179Y1A0524@ksrmce.ac.in
25	179Y1A0525	Chavva Sravani	B.Tech V SEM	179Y1A0525@ksrmce.ac.in
26	179Y1A0526	C.V.S.Kowshiknath Reddy	B.Tech V SEM	179Y1A0526@ksrmce.ac.in
27	179Y1A0527	Chenuru Brahmani	B.Tech V SEM	179Y1A0527@ksrmce.ac.in
28	179Y1A0528	C Shaik Mahammed Aslam	B.Tech V SEM	179Y1A0528@ksrmce.ac.in
29	179Y1A0529	Chilakala Mahammadrafi	B.Tech V SEM	179Y1A0529@ksrmce.ac.in
30	179Y1A0530	C.Maheswari	B.Tech V SEM	179Y1A0530@ksrmce.ac.in

31	179Y1A0531	Chinthala Akhila	B.Tech V SEM	179Y1A0531@ksrmce.ac.in
32	179Y1A0532	D Mohammmad Sharuk	B.Tech V SEM	179Y1A0532@ksrmce.ac.in
33	179Y1A0533	Dabbera Sowmya	B.Tech V SEM	179Y1A0533@ksrmce.ac.in
34	179Y1A0534	Daddanala Yogya Vignesh	B.Tech V SEM	179Y1A0534@ksrmce.ac.in
35	179Y1A0535	Dandu Sai Kiran	B.Tech V SEM	179Y1A0535@ksrmce.ac.in
36	179Y1A0536	Dasari Venkateswarlu	B.Tech V SEM	179Y1A0536@ksrmce.ac.in
37	179Y1A0537	D. Yogesh	B.Tech V SEM	179Y1A0537@ksrmce.ac.in
38	179Y1A0538	D Anusha	B.Tech V SEM	179Y1A0538@ksrmce.ac.in
39	179Y1A0539	Doddi Devananda	B.Tech V SEM	179Y1A0539@ksrmce.ac.in
40	179Y1A0540	D.Yasaswini	B.Tech V SEM	179Y1A0540@ksrmce.ac.in
41	179Y1A0541	Donthu Sukanya	B.Tech V SEM	179Y1A0541@ksrmce.ac.in
42	179Y1A0542	Draksham Naveen	B.Tech V SEM	179Y1A0542@ksrmce.ac.in
43	179Y1A0543	D. Jyothsna	B.Tech V SEM	179Y1A0543@ksrmce.ac.in
44	179Y1A0544	E. Abhishek Kumar Sahu	B.Tech V SEM	179Y1A0544@ksrmce.ac.in
45	179Y1A0545	E.Prudhavi	B.Tech V SEM	179Y1A0545@ksrmce.ac.in
46	179Y1A0546	G. Mamatha	B.Tech V SEM	179Y1A0546@ksrmce.ac.in
47	179Y1A0547	G. Sudeesh Chandra Rao	B.Tech V SEM	179Y1A0547@ksrmce.ac.in
48	179Y1A0548	G. Haritha	B.Tech V SEM	179Y1A0548@ksrmce.ac.in
49	179Y1A0549	G. Vinod Kumar	B.Tech V SEM	179Y1A0549@ksrmce.ac.in
50	179Y1A0550	G. Sai Kiran Reddy	B.Tech V SEM	179Y1A0550@ksrmce.ac.in
51	179Y1A0551	G. Ajay Kumar	B.Tech V SEM	179Y1A0551@ksrmce.ac.in
52	179Y1A0552	G. Rajavardhan Reddy	B.Tech V SEM	179Y1A0552@ksrmce.ac.in
53	179Y1A0553	G. Sai Sujitha Reddy	B.Tech V SEM	179Y1A0553@ksrmce.ac.in
54	179Y1A0554	P. Harini	B.Tech V SEM	179Y1A0554@ksrmce.ac.in
55	179Y1A0555	I Manoj Kumar	B.Tech V SEM	179Y1A0555@ksrmce.ac.in
56	179Y1A0556	I Ram Bhupal Reddy	B.Tech V SEM	179Y1A0556@ksrmce.ac.in
57	179Y1A0557	J. Haritha	B.Tech V SEM	179Y1A0557@ksrmce.ac.in
58	179Y1A0558	J Sai Kumar	B.Tech V SEM	179Y1A0558@ksrmce.ac.in
59	179Y1A0559	K. Ramacharan	B.Tech V SEM	179Y1A0559@ksrmce.ac.in
60	179Y1A0560	K Kavitha Reddy	B.Tech V SEM	179Y1A0560@ksrmce.ac.in
61	179Y1A0561	K Varshitha Reddy	B.Tech V SEM	179Y1A0561@ksrmce.ac.in
62	179Y1A0562	KVenkata Sai Kumar Reddy	B.Tech V SEM	179Y1A0562@ksrmce.ac.in
63	179Y1A0563	Kamineni Keerthana	B.Tech V SEM	179Y1A0563@ksrmce.ac.in
64	179Y1A0564	Kancharla Sasikala	B.Tech V SEM	179Y1A0564@ksrmce.ac.in
65	179Y1A0565	K Venakata Subba Reddy	B.Tech V SEM	179Y1A0565@ksrmce.ac.in
66	179Y1A0566	Kanchaveerla Harika	B.Tech V SEM	179Y1A0566@ksrmce.ac.in
67	179Y1A0567	K.Siva Narayana	B.Tech V SEM	179Y1A0567@ksrmce.ac.in
68	179Y1A0568	Kotluru Noor Basha	B.Tech V SEM	179Y1A0568@ksrmce.ac.in
69	179Y1A0569	Kuntavala Pruthvi	B.Tech V SEM	179Y1A0569@ksrmce.ac.in
70	179Y1A0570	Kuruba Keerthi	B.Tech V SEM	179Y1A0570@ksrmce.ac.in

Coordinator

HoD

Dr. M. Sreenivasulu,
M. E., Ph. D.
Professer & HOD CSE
K.S.R.M. College of Engineering
K A D A P A - 516 003

K.S.R.M. College of Engineering (Autonomous), Kadapa.

Department of CSE

Android Application Development (with Kotlin)

Module 1:

Getting Started with Kotlin: Introduction to Kotlin, Benefits of using Kotlin, Use Kotlin REPL to practice basic expressions, Control flow statements in Kotlin, Null safety with Kotlin, Learn about Kotlin REPL, operators, types, conditional statements, null safety, arrays, lists and loops in Kotlin.

Functions in Kotlin: Creating and calling functions with default and named arguments, Writing concise and compact functions, Passing functions as arguments to other functions.

Module 2:

Object Oriented Programming: Introduction to object-oriented programming in Kotlin, Classes and objects in Kotlin, Constructors, Visibility modifiers, Subclasses and inheritance, Interfaces, Data classes, Singleton class enums, Pairs, triples and collections in Kotlin, Extensions in Kotlin

Module 3:

Introduction to Android: Installing Android Studio, Creating an Android app project, Deploying the app to an emulator or a device, Building an Android app that contains images and a click handler, Modifying views within the layout of an app, Adding libraries to module gradle file, Creating Layouts and working with Constraint Layout.

App Navigation: Create a Fragment, Define navigation paths, Start an External Activity, Activity and Fragments LifeCycles.

Module 4:

RecyclerView: RecyclerView fundamentals, DifUtil and Data Binding with RecyclerView, GridLayout with RecyclerView. Interacting with RecyclerView Items.Headers in RecyclerView.

Connect to the Internet: Getting Data from the Internet, Loading and Displaying images from the Internet, Filtering and Details Views with Internet Data.

Textbooks:

1. Android Application Development with Kotlin: Build Your First Android App in No Time by Hardik Trivedi, bpb.
2. Android Development With Kotlin by Marcin Moskala, Igor Wojda, PACKT.

Reference Textbook:

1. Kotlin for Android App Development by Paperback, Peter Sommerhoff, Bernhard Rumpe, Pearson Education.

K.S.R.M. COLLEGE OF ENGINEERING

(UGC - AUTONOMOUS)

Kadapa, Andhra Pradesh,India - 516003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution

Department of Computer Science & Engineering

Certificate Course on Android Application Development from 09/09/2019 to 27/09/2019

Schedule

S.No	Date	Time	Faculty	Topic
1	09/09/2019	3:00 PM to 6.00 PM	Smt. S. Riyaz Bhanu Sri. G. Nagendra Babu Sri. S. Khaja Khizar	Inauguration,Getting Started with Kotlin: Introduction to Kotlin, Benefits of using Kotlin, Use Kotlin REPL to practice basic expressions
2	10/09/2019	4:00 PM to 6.00 PM	Sri. G. Nagendra Babu	Null safety with Kotlin, Learn about Kotlin REPL
3	11/09/2019	3:00 PM to 6.00 PM	Sri. G. Nagendra Babu	operators, tpressions, ypes, conditional statements, null safety, arrays, lists and loops in Kotlin.
4	12/09/2019	4:00 PM to 6.00 PM	Sri. S. Khaja Khizar	Functions in Kotlin: Creating and calling functions with default and named arguments
5	13/09/2019	3:00 PM to 6.00 PM	Sri. S. Khaja Khizar	Writing concise and compact functions, Passing functions as arguments to other functions.
6	14/09/2019	4:00 PM to 6.00 PM	Sri. S. Khaja Khizar	Object Oriented Programming: Introduction to object-oriented programming in Kotlin, Classes and objects in Kotlin
7	16/09/2019	4:00 PM to 6.00 PM	Sri. S. Khaja Khizar	Constructors, Visibility modifiers, Subclasses and inheritance, Interfaces, ,

8	17/09/2019	4:00 PM to 6.00 PM	Sri. S. Khaja Khizar	Data classes, Singleton class enums, Pairs
9	18/09/2019	4:00 PM to 6.00 PM	Sri. S. Khaja Khizar	triples and collections in Kotlin, Extensions in Kotlin
10	19/09/2019	4:00P M to 6.00 PM	Sri. S. Khaja Khizar	Introduction to Android: Installing Android Studio,
11	20/09/19	4:00PM to 6.00PM	Sri. S. Khaja Khizar	Creating an Android app project, Deploying the app to an emulator or a device,
12	21/09/19	4:00PM to 6.00PM	Sri. S. Khaja Khizar	Building an Android app that contains images and a click handler, Modifying views within the layout of an app,
13	23/09/2019	4:00 PM to 6.00 PM	Sri. S. Khaja Khizar	Adding libraries to module gradle file, Creating Layouts and working with Constraint Layout.
14	24/09/2019	4:00 PM to 6.00 PM	Sri. S. Khaja Khizar	App Navigation: Create a Fragment,
15	25/09/2019	4:00 PM to 6.00 PM	Sri. S. Khaja Khizar	Define navigation paths, Start an External Activity, Activity and Fragments LifeCycles.
16	26/09/2019	4:00 PM to 6.00 PM	Sri. S. Khaja Khizar	RecyclerView: RecyclerView fundamentals, DifUtil
17	27/09/2019	4:00 PM to 6.00 PM	Smt. S. Riyaz Bhanu Sri. G. Nagendra Babu Sri. S. Khaja Khizar	Exam, Certificate distribution

Riyaz Bhanu
Coordinator

HOD

Dr. M. Sreenivasulu,
M. E., Ph. D.
Professor & HOD CSE
K.S.R.M. College of Engineering
KADAPA - 516 003

K.S.R.M. COLLEGE OF ENGINEERING

(UGC - AUTONOMOUS)

Kadapa, Andhra Pradesh, India - 516003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution

Department of Computer Science & Engineering

Certificate Course on Android Application Development

Attendance Sheet

S.No	Roll Num	Name Of The Student	09/09/2019	10/09/2019	11/09/2019	12/09/2019	13/09/19	14/09/2019	16/09/2019	17/09/2019	18/09/2019	19/09/2019	20/09/2019	21/09/2019	23/09/2019	24/09/2019	25/09/2019	26/09/2019	27/09/2019
1	169Y1A0502	Aakuleti Sreekanth	P	P	P	P	P	P	P	A	P	P	P	P	P	P	P	P	P
2	169Y1A0503	Abburi Vanitha (W)	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P
3	169Y1A0504	Alamuru Shireesha (W)	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P
4	169Y1A0505	Alamuru Venkata Ravindra Reddy	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P
5	169Y1A0507	Avula Subrahmanya Sumanth Kumar	P	P	P	P	P	P	P	P	P	P	P	P	P	P	A	P	P
6	169Y1A0509	Bandla Venkatesh	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P
7	169Y1A0510	Banka Govardhan Reddy	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P
8	169Y1A0511	Batthini Anandsrinivasyadav	P	A	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P
9	169Y1A0514	Boggarapu Nagamanasa (W)	P	P	P	A	P	P	P	P	P	P	P	P	P	P	P	P	P
10	169Y1A0515	Bojja Bhagya Sree (W)	P	P	A	P	P	P	P	P	P	P	A	P	P	P	P	P	P
11	169Y1A0523	Chukka Swapna (W)	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P
12	169Y1A0524	Dalaie Vasudha (W)	P	P	P	P	A	P	P	P	P	P	P	P	P	P	P	P	P
13	169Y1A0549	Ketham Vara Lakshmi (W)	P	P	P	P	P	A	P	P	A	P	P	P	P	P	P	P	P
14	169Y1A0550	Kokatam Maheswarreddy	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P
15	169Y1A0551	Konduru Babitha (W)	P	P	P	P	P	P	P	P	P	P	P	A	P	P	P	P	P
16	169Y1A0576	Palempalli Veeraprasanna (W)	P	P	P	P	P	P	P	P	P	P	A	P	P	P	P	A	P

17	169Y1A0577	Palleti Venkata Yeseshwini (W)	P	P	P	P	P	P	P	A	P	P	A	P	P	P	P	P	P	P
18	169Y1A0578	Pamidi Gowtham Reddy	P	A	P	P	P	P	P	P	P	P	P	P	P	P	P	A	P	P
19	169Y1A0579	Pappireddy Vijitha (W)	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P
20	179Y1A0520	Bosigari Bharathi	P	P	P	P	P	A	P	P	P	P	P	P	P	P	P	P	P	P
21	179Y1A0521	Bukkarayasamudram Ramana Sai	P	P	A	P	P	P	P	P	P	P	P	A	P	P	P	P	P	P
22	179Y1A0522	C.Lohith Kumar	P	P	P	P	P	P	P	P	P	P	A	P	P	P	P	A	P	P
23	179Y1A0523	Chanda Harinadh Reddy	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P
24	179Y1A0524	Chappidi Harsha Vardhan Reddy	P	P	P	P	P	P	P	P	P	A	P	P	P	P	P	P	P	P
25	179Y1A0525	Chavva Sravani	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P
26	179Y1A0526	C.V.S.Kowshiknath Reddy	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P
27	179Y1A0527	Chenuru Brahmani	P	P	P	P	A	P	P	P	P	P	P	P	P	P	P	P	P	P
28	179Y1A0528	Cheppali Shaik Mahammed Aslam	P	P	P	P	P	P	P	P	P	A	A	P	P	P	P	P	P	P
29	179Y1A0529	Chilakala Mahammadrafi	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P
30	179Y1A0530	C.Maheswari	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P
31	179Y1A0531	Chinthala Akhila	P	P	P	A	P	P	P	P	P	P	P	P	P	P	P	P	P	P
32	179Y1A0532	D Mohammmad Sharuk	P	P	P	P	P	A	P	P	A	P	P	P	P	P	P	P	P	P
33	179Y1A0533	Dabbera Sowmya	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P
34	179Y1A0534	Daddanala Yogya Vignesh	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P
35	179Y1A0535	Dandu Sai Kiran	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P
36	179Y1A0536	Dasari Venkateswarlu	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	A	P	P
37	179Y1A0537	D. Yogesh	P	P	P	P	P	P	P	A	P	P	P	P	P	P	P	P	P	P
38	179Y1A0538	D Anusha	P	P	P	P	P	P	P	P	P	P	P	A	P	P	P	A	P	P
39	179Y1A0539	Doddi Devananda	P	P	P	P	P	P	P	P	P	A	P	P	P	P	P	P	P	P
40	179Y1A0540	D.Yasaswini	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P
41	179Y1A0541	Donthu Sukanya	P	P	P	P	P	P	P	P	P	P	A	P	P	P	P	P	P	P
42	179Y1A0542	Draksham Naveen	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P
43	179Y1A0543	D. Jyothsna	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P
44	179Y1A0544	E. Abhishek Kumar Sahu	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P
45	179Y1A0545	E.Prudhavi	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P
46	179Y1A0546	G. Mamatha	P	P	P	P	P	A	P	P	P	P	P	P	P	A	P	P	P	P
47	179Y1A0547	G. Sudeesh Chandra Rao	P	A	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P
48	179Y1A0548	G. Haritha	P	P	A	P	P	P	P	P	A	P	P	P	P	P	P	P	P	P

49	179Y1A0549	G. Vinod Kumar	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	A	P
50	179Y1A0550	G. Sai Kiran Reddy	P	P	P	P	P	A	P	P	P	P	P	P	P	P	P	P	P
51	179Y1A0551	G. Ajay Kumar	P	P	P	P	P	P	P	A	P	P	P	P	P	P	P	P	P
52	179Y1A0552	G. Rajavardhan Reddy	P	P	P	P	P	P	A	P	P	A	P	P	P	P	P	P	P
53	179Y1A0553	G. Sai Sujitha Reddy	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P
54	179Y1A0554	P. Harini	P	P	P	P	P	P	P	P	P	P	A	P	P	P	P	P	P
55	179Y1A0555	I Manoj Kumar	P	P	P	P	P	P	P	P	P	P	A	P	P	P	A	P	P
56	179Y1A0556	I Ram Bhupal Reddy	P	P	P	P	P	P	P	P	A	P	P	P	P	P	P	P	P
57	179Y1A0557	J. Haritha	P	P	P	P	P	P	P	A	P	P	P	P	P	P	P	P	P
58	179Y1A0558	J Sai Kumar	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P
59	179Y1A0559	K. Ramacharan	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P
60	179Y1A0560	K Kavitha Reddy	P	P	P	P	P	P	P	P	P	P	P	P	A	P	P	P	P
61	179Y1A0561	K Varshitha Reddy	P	P	P	P	P	P	P	P	A	P	P	P	P	P	P	P	P
62	179Y1A0562	Kambham Venkata Sai Kumar Reddy	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P
63	179Y1A0563	Kamineni Keerthana	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P
64	179Y1A0564	Kancharla Sasikala	P	P	P	P	P	P	P	P	P	P	P	A	P	P	P	P	P
65	179Y1A0565	Kancharla Venakata Subba Reddy	P	A	P	P	P	P	P	P	P	P	P	P	P	P	A	P	P
66	179Y1A0566	Kanchaveerla Harika	P	P	P	P	P	A	P	P	P	P	P	P	P	P	P	P	P
67	179Y1A0567	K.Siva Narayana	P	P	P	P	P	P	P	P	A	P	P	P	P	P	P	P	P
68	179Y1A0568	Kotluru Noor Basha	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P
69	179Y1A0569	Kuntavala Pruthvi	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P	P
70	179Y1A0570	Kuruba Keerthi	P	P	P	P	P	P	P	P	P	P	P	A	P	P	P	P	P

Riyaz Bamy
Coordinator

HOD

Dr. M. Sreenivasulu,
M.E., Ph.D.
Professor & HOD CSE
K.S.R.M. College of Engineering
KADAPA - 516 003

K.S.R.M. COLLEGE OF ENGINEERING

(UGC - Autonomous)

Kadapa, Andhra Pradesh, India– 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

K.S.R.M. COLLEGE OF ENGINEERING

(UGC-AUTONOMOUS)

Kadapa, Andhra Pradesh, India– 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution

ACTIVITY REPORT

Certification Course
On
"Android Application Development"

09-09-2019 to 27-09-2019

Target Group	:	**B.Tech V Semester & VII Semester CSE Students**
Details of Participants	:	**70 Students**
Coordinator	:	**Mrs. S. Riyaz Bhanu** **Asst. Prof, Dept. of CSE,KSRMCE**
Organizing Department	:	**Department of Computer Science & Engineering**
Venue	:	**MB209 (MAD Lab)**

Description:

Certification course on "Android Application Development" was organized by Dept. of CSE from **09-09-2019 to 27-09-2019** in MB 209(MAD LAB). Mrs. S. Riyaz Bhanu acted as Course Coordinator & resource persons are Mr. G. Nagendra Babu and Mr. S. Khaja Khizar. This course will clear up the fundamental concepts of Android Application Development (with Kotlin). **Kotlin** helped their teams become more productive and write higher quality apps. Write better **Android** apps faster with **Kotlin**. Thirty Seven hours course was successfully completed and participation certificates were provided to the participants.

Event Photos:

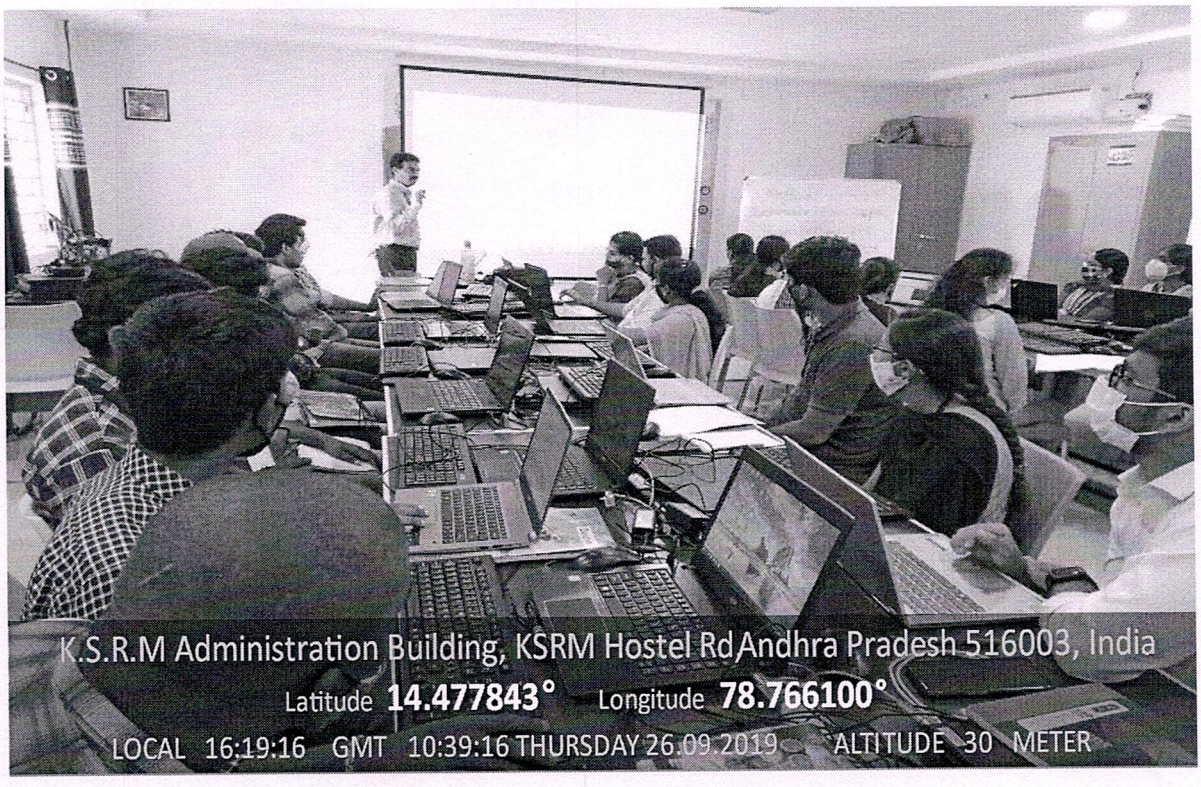
K.S.R.M. COLLEGE OF ENGINEERING

(UGC - Autonomous)
Kadapa, Andhra Pradesh, India- 516 003
Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Event banner

Coordinator gives brief overview about certificate course (Android Application Development)



Students participated in practical session

Riyaz Bany
Coordinator

HoD
Dr. M. Sreenivasulu,
M. E., Ph. D.
Professer & HOD CSE
K.S.R.M. College of Engineering
K A D A P A - 516 003

K.S.R.M. COLLEGE OF ENGINEERING

(UGC - Autonomous)

Kadapa, Andhra Pradesh, India– 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE OF PARTICIPATION

This is to certify that Mr/Miss. Konduru Babitha bearing Roll Number. 169Y1A0551 participated in a certification course on "**Android Application Development**" organized by department of Computer Science and Engineering from 09-09-2019 to 27-09-2019.

Riyaz Bamy		V. S. S. Murthy
COORDINATOR	HOD	PRINCIPAL

K.S.R.M. COLLEGE OF ENGINEERING

(UGC - Autonomous)
Kadapa, Andhra Pradesh, India– 516 003
Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE OF PARTICIPATION

This is to certify that Mr/Miss. G. Hasitha bearing Roll Number. 179Y1A0548 participated in a certification course on "**Android Application Development**" organized by department of Computer Science and Engineering from 09-09-2019 to 27-09-2019.

Riyaz Bamy		V. S. S. Murty
COORDINATOR	HOD	PRINCIPAL

K.S.R.M. COLLEGE OF ENGINEERING

(UGC - Autonomous)
Kadapa, Andhra Pradesh, India– 516 003
Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE OF PARTICIPATION

This is to certify that Mr/Miss. Kuruba Keerthi bearing Roll Number. 179Y1A0570 participated in a certification course on "**Android Application Development**" organized by department of Computer Science and Engineering from 09-09-2019 to 27-09-2019.

Riyaz Bamy		V. S. S. Murty
COORDINATOR	HOD	PRINCIPAL

K.S.R.M. COLLEGE OF ENGINEERING

UGC - AUTONOMOUS

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

Kadapa, Andhra Pradesh, India– 516 003

FEEDBACK FORM

Certificate Course on "Android Application Development", from 09-09-2019 to 27-09-2019

Organized

by

Department of Computer Science & Engineering

NAME:

Roll No:

S.No	Feedback Item	Excellent	Very Good	Good	Average	Below Average
1	Organization of certificate course and session planning by resource person.					
2	Clarity in content delivery.					
3	Content is relevant and useful.					
4	Adequate opportunity to interact with resource person.					
5	Judicious mix of concepts, principles and practices.					
6	Assignments and tasks are interesting and challenging.					

Any suggestions for improvement.

Signature

K.S.R.M. COLLEGE OF ENGINEERING

UGC - AUTONOMOUS

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

Kadapa, Andhra Pradesh, India– 516 003

FEEDBACK FORM

Certificate Course on "Android Application Development", from 09-09-2019 to 27-09-2019

Organized

by

Department of Computer Science & Engineering

NAME: A. Sreekanth

Roll No: 169Y1A0502

S.No	Feedback Item	Excellent	Very Good	Good	Average	Below Average
1	Organization of certificate course and session planning by resource person.	✓				
2	Clarity in content delivery.	✓				
3	Content is relevant and useful.		✓			
4	Adequate opportunity to interact with resource person.		✓			
5	Judicious mix of concepts, principles and practices.			✓		
6	Assignments and tasks are interesting and challenging.	✓				

Any suggestions for improvement.

—

Signature

K.S.R.M. COLLEGE OF ENGINEERING

UGC - AUTONOMOUS

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

Kadapa, Andhra Pradesh, India– 516 003

FEEDBACK FORM

Certificate Course on "Android Application Development", from 09-09-2019 to 27-09-2019

Organized

by

Department of Computer Science & Engineering

NAME: Abburi vanitha

Roll No: 16941A0503

S.No	Feedback Item	Excellent	Very Good	Good	Average	Below Average
1	Organization of certificate course and session planning by resource person.	✓				
2	Clarity in content delivery.	✓				
3	Content is relevant and useful.		✓			
4	Adequate opportunity to interact with resource person.	✓				
5	Judicious mix of concepts, principles and practices.	✓				
6	Assignments and tasks are interesting and challenging.				✓	

Any suggestions for improvement.

—

Signature

K.S.R.M. COLLEGE OF ENGINEERING

UGC - AUTONOMOUS

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

Kadapa, Andhra Pradesh, India– 516 003

FEEDBACK FORM

Certificate Course on "Android Application Development", from 09-09-2019 to 27-09-2019

Organized

by

Department of Computer Science & Engineering

NAME: 169Y1A0507, A. subramanya. sumanth kumar

Roll No: 169Y1A0507

S.No	Feedback Item	Excellent	Very Good	Good	Average	Below Average
1	Organization of certificate course and session planning by resource person.	✓				
2	Clarity in content delivery.		✓			
3	Content is relevant and useful.			✓		
4	Adequate opportunity to interact with resource person.		✓			
5	Judicious mix of concepts, principles and practices.	✓				
6	Assignments and tasks are interesting and challenging.		✓			

Any suggestions for improvement.

(K)

Signature

K.S.R.M. COLLEGE OF ENGINEERING

UGC - AUTONOMOUS

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

Kadapa, Andhra Pradesh, India– 516 003

FEEDBACK FORM

Certificate Course on "Android Application Development", from 09-09-2019 to 27-09-2019

Organized

by

Department of Computer Science & Engineering

NAME: Chawa Sravani

Roll No: 179Y1A0525

S.No	Feedback Item	Excellent	Very Good	Good	Average	Below Average
1	Organization of certificate course and session planning by resource person.		✓			
2	Clarity in content delivery.	✓				
3	Content is relevant and useful.		✓			
4	Adequate opportunity to interact with resource person.			✓		
5	Judicious mix of concepts, principles and practices.	✓				
6	Assignments and tasks are interesting and challenging.		✓			

Any suggestions for improvement.

Sravani
Signature

Lesson 2: Functions

About this lesson

Lesson 2: Functions

Programs in Kotlin

Setting up

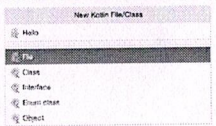
Before you can write code and run programs, you need to:

- Create a file in your project
- Create a `main()` function
- Pass arguments to `main()` (Optional)
- Use any passed arguments in function calls (Optional)
- Run your program

Create a new Kotlin file

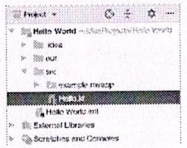
In IntelliJ IDEA's Project pane, under **Hello World**, right-click the `src` folder.

- Select **New > Kotlin File/Class**.
- Select **File**, name the file `Hello`, and press **Enter**.



Create a Kotlin file

You should now see a file in the `src` folder called `Hello.kt`.



Create a main() function

`main()` is the entry point for execution for a Kotlin program.

In the `Hello.kt` file:

```
fun main(args: Array<String>) {
    println("Hello, world!")
}
```

The args in the `main()` function are optional.

Run your Kotlin program

To run your program, click the Run icon (▶) to the left of the `main()` function.

```
fun main(args: Array<String>) {
    println("Hello, world!")
}
```

IntelliJ IDEA runs the program, and displays the results in the console.

```
/Library/Java/JavaVirtualMachines/jdk-13.0.2.jdk/Contents/Home/bin/java ...
Hello, world!

Process finished with exit code 0
```

Pass arguments to main()

Select **Run > Edit Configurations** to open the **Run/Debug Configurations** window.

Use arguments in `main()`

Use `args[0]` to access the first input argument passed to `main()`.

```
fun main(args: Array<String>) {
    println("Hello, ${args[0]}")
}

=> Hello, Kotlin!
```

(Almost) Everything has a value

(Almost) Everything is an expression

In Kotlin, almost everything is an expression and has a value. Even an `if` expression has a value.

```
val temperature = 20
val isHot = if (temperature > 40) true else false
println(isHot)
=> false
```

Expression values

Sometimes, that value is `kotlin.Unit`.

```
val isUnit = println("This is an expression")
println(isUnit)

=> This is an expression
   kotlin.Unit
```

Functions in Kotlin

About functions

- A block of code that performs a specific task
- Breaks a large program into smaller modular chunks
- Declared using the `fun` keyword
- Can take arguments with either named or default values

Parts of a function

Earlier, you created a simple function that printed "Hello World".

```
fun printHello() {
    println("Hello World")
}

printHello()
```

Unit returning functions

If a function does not return any useful value, its return type is `Unit`.

```
fun printHello(name: String?): Unit {
    println("Hi there!")
}
```

`Unit` is a type with only one value: `Unit`.

Unit returning functions

The `Unit` return type declaration is optional.

```
fun printHello(name: String?): Unit {
    println("Hi there!")
}
```

is equivalent to:

```
fun printHello(name: String?) {
    println("Hi there!")
}
```


Function arguments

Functions may have:

- Default parameters
- Required parameters
- Named arguments

Default parameters

Default values provide a fallback if no parameter value is passed.

```
fun drive(speed: String = "fast") {
    println("driving $speed")
}
```

Use "=" after the type to define default values

```
drive() ⇒ driving fast
drive("slow") ⇒ driving slow
drive(speed = "turtle-like") ⇒ driving turtle-like
```


Required parameters

If no default is specified for a parameter, the corresponding argument is required.

Required parameters

```
fun tempToday(day: String, temp: Int) {
    println("Today is $day and it's $temp degrees.")
}
```


Default versus required parameters

Functions can have a mix of default and required parameters.

```
fun reformat(str: String,
             divideByCamelHumps: Boolean,
             wordSeparator: Char,
             normalizeCase: Boolean = true){
```

Has default value

Pass in required arguments.

```
reformat("Today is a day like no other day", false, '_')
```


Named arguments

To improve readability, use named arguments for required arguments.

```
reformat(str, divideByCamelHumps = false, wordSeparator = '_')
```

It's considered good style to put default arguments after positional arguments, that way callers only have to specify the required arguments.

Compact functions

Single-expression functions

Compact functions, or single-expression functions, make your code more concise and readable.

```
fun double(x: Int): Int {
    x * 2
}
```

Complete version

```
fun double(x: Int):Int = x * 2
```

Compact version

Lambdas and higher-order functions

Kotlin functions are first-class

- Kotlin functions can be stored in variables and data structures
- They can be passed as arguments to, and returned from, other higher-order functions
- You can use higher-order functions to create new "built-in" functions

Lambda functions

A lambda is an expression that makes a function that has no name.

Parameter and type

Function arrow

```
var dirtLevel = 20
val waterFilter = {level: Int -> level / 2}
println(waterFilter(dirtLevel))
// 10
```

Code to execute

Syntax for function types

Kotlin's syntax for function types is closely related to its syntax for lambdas. Declare a variable that holds a function.

```
val waterFilter: (Int) -> Int = {level -> level / 2}
```

Variable name

Data type of variable (function type)

Function

Higher-order functions

Higher-order functions take functions as parameters, or return a function.

```
fun encodeMsg(msg: String, encode: (String) -> String): String
{
    return encode(msg)
}
```

The body of the code calls the function that was passed as the second argument, and passes the first argument along to it.

Higher-order functions

To call this function, pass it a string and a function.

```
val enc1: (String) -> String = { input -> input.toUpperCase() }
println(encodeMsg("abc", enc1))
```

Using a function type separates its implementation from its usage.

Passing a function reference

Use the `::` operator to pass a named function as an argument to another function.

```
fun enc2(input:String): String = input.reversed()
encodeMessage("abc", ::enc2)
```

Passing a named function, not a lambda

The `::` operator lets Kotlin know that you are passing the function reference as an argument, and not trying to call the function.

Last parameter call syntax

Kotlin prefers that any parameter that takes a function is the last parameter.

```
encodeMessage("acronym", { input -> input.toUpperCase() })
```

You can pass a lambda as a function parameter without putting it inside the parentheses.

```
encodeMsg("acronym") { input -> input.toUpperCase() }
```


Using higher-order functions

Many Kotlin built-in functions are defined using last parameter call syntax.

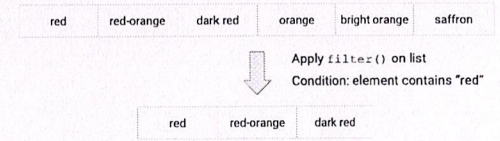
```
inline fun repeat(times: Int, action: (Int) -> Unit)
```

```
repeat(3) {
    println("Hello")
}
```


List filters

List filters

Get part of a list based on some condition



Iterating through lists

If a function literal has only one parameter, you can omit its declaration and the "->". The parameter is implicitly declared under the name `it`.

```
val ints = listOf(1, 2, 3)
ints.filter { it > 0 }
```

Filter iterates through a collection, where `it` is the value of the element during the iteration. This is equivalent to:

```
ints.filter { n: Int -> n > 0 }   OR   ints.filter { n -> n > 0 }
```


List filters

The filter condition in curly braces `{}` tests each item as the filter loops through. If the expression returns `true`, the item is included.

```
val books = listOf("nature", "biology", "birds")
println(books.filter { it[0] == 'b' })
```

```
=> [biology, birds]
```


Eager and lazy filters

Evaluation of expressions in lists:

- **Eager:** occurs regardless of whether the result is ever used
- **Lazy:** occurs only if necessary at runtime

Lazy evaluation of lists is useful if you don't need the entire result, or if the list is exceptionally large and multiple copies wouldn't wouldn't fit into RAM.

Eager filters

Filters are eager by default. A new list is created each time you use a filter.

```
val instruments = listOf("viola", "cello", "violin")
val eager = instruments.filter { it [0] == 'v' }
println("eager: " + eager)
```

```
=> eager: [viola, violin]
```


Lazy filters

Sequences are data structures that use lazy evaluation, and can be used with filters to make them lazy.

```
val instruments = listOf("viola", "cello", "violin")
val filtered = instruments.asSequence().filter { it[0] == 'v'}
println("filtered: " + filtered)
```

```
=> filtered: kotlin.sequences.FilteringSequence@3f86cc1c4
```


Sequences -> lists

Sequences can be turned back into lists using `toList()`.

```
val filtered = instruments.asSequence().filter { it[0] == 'v'}
val newList = filtered.toList()
println("new list: " + newList)
```

```
=> new list: [viola, violin]
```


Other list transformations

- `map()` performs the same transform on every item and returns the list.

```
val numbers = setOf(1, 2, 3)
println(numbers.map { it * 3 })
=> [3, 6, 9]
```

- `flatten()` returns a single list of all the elements of nested collections.

```
val numberSets = listOf(setOf(1, 2, 3), setOf(4, 5), setOf(1, 2))
println(numberSets.flatten())
=> [1, 2, 3, 4, 5, 1, 2]
```


Summary

Summary

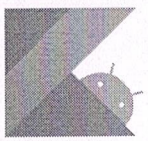
In Lesson 2, you learned how to:

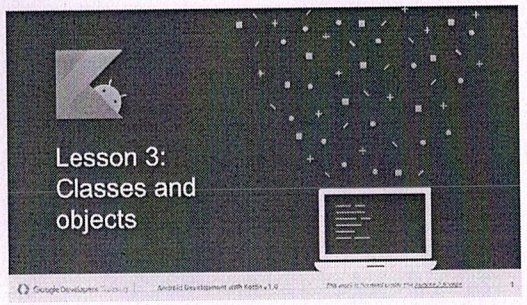
- Create a file and a `main()` function in your project, and run a program
- Pass arguments to the `main()` function
- Use the returned value of an expression
- Use default arguments to replace multiple versions of a function
- Use compact functions, to make code more readable
- Use lambdas and higher-order functions
- Use eager and lazy list filters

Pathway

Practice what you've learned by completing the pathway:

Lesson 2: Functions





About this lesson

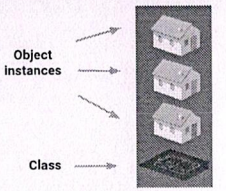
Lesson 3: Classes and objects

- Classes
- Inheritance
- Extension functions
- Special classes
- Organizing your code
- Summary

Classes

Class

- Classes are blueprints for objects
- Classes define methods that operate on their object instances



Class versus object instance

House Class

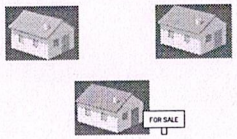
Data

- House color (String)
- Number of windows (Int)
- Is for sale (Boolean)

Behavior

- `updateColor()`
- `putOnSale()`

Object Instances



Define and use a class

Class Definition

```
class House {
  val color: String = "white"
  val numberOfWindows: Int = 2
  val isForSale: Boolean = false

  fun updateColor(newColor: String){...}
  ...
}
```

Create New Object Instance

```
val myHouse = House()
println(myHouse)
```

Constructors

When a constructor is defined in the class header, it can contain:

- No parameters

  `class A`
- Parameters
  - Not marked with `var` or `val` → copy exists only within scope of the constructor

    `class B(x: Int)`
  - Marked `var` or `val` → copy exists in all instances of the class

    `class C(val y: Int)`

Constructor examples

```
class A
```

```
val aa = A()
```

```
class B(x: Int)
```

```
val bb = B(12)
println(bb.x)
=> compiler error unresolved
reference
```

```
class C(val y: Int)
```

```
val cc = C(42)
println(cc.y)
=> 42
```

Default parameters

Class instances can have default values.

- Use default values to reduce the number of constructors needed
- Default parameters can be mixed with required parameters
- More concise (don't need to have multiple constructor versions)

```
class Box(val length: Int, val width:Int = 20, val height:Int = 40)

val box1 = Box(100, 20, 40)
val box2 = Box(length = 100)
val box3 = Box(length = 100, width = 20, height = 40)
```

Primary constructor

Declare the primary constructor within the class header.

```
class Circle(i: Int) {
    init {
        ...
    }
}
```

This is technically equivalent to:

```
class Circle {
    constructor(i: Int) {
        ...
    }
}
```

Initializer block

- Any required initialization code is run in a special `init` block
- Multiple `init` blocks are allowed
- `init` blocks become the body of the primary constructor

Initializer block example

Use the `init` keyword:

```
class Square(val side: Int) {
    init {
        println(side * 2)
    }
}

val s = Square(10)
=> 20
```

Multiple constructors

- Use the `constructor` keyword to define secondary constructors
- Secondary constructors must call:
  - The primary constructor using `this` keyword

    OR
  - Another secondary constructor that calls the primary constructor
- Secondary constructor body is not required

Multiple constructors example

```
class Circle(val radius:Double) {
    constructor(name:String) : this(1.0)
    constructor(diameter:Int) : this(diameter / 2.0) {
        println("in diameter constructor")
    }
    init {
        println("Area: ${Math.PI * radius * radius}")
    }
}
val c = Circle(3)
```

Properties

- Define properties in a class using `val` or `var`
- Access these properties using dot `.` notation with property name
- Set these properties using dot `.` notation with property name (only if declared a `var`)

Person class with name property

```
class Person(var name: String)
fun main() {
    val person = Person("Alex")
    println(person.name)        <-- Access with .<property name>
    person.name = "Joey"        <-- Set with .<property name>
    println(person.name)
}
```

Custom getters and setters

If you don't want the default `get`/`set` behavior:

- Override `get()` for a property
- Override `set()` for a property (if defined as a `var`)

Format:
```
var propertyName: DataType = initialValue
    get() = ...
    set(value) {
        ...
    }
```


Custom getter

```
class Person(val firstName: String, val lastName:String) {
    val fullName:String
        get() {
            return "$firstName $lastName"
        }
}

val person = Person("John", "Doe")
println(person.fullName)
=> John Doe
```


Custom setter

```
var fullName:String = ""
    get() = "$firstName $lastName"
    set(value) {
        val components = value.split(" ")
        firstName = components[0]
        lastName = components[1]
        field = value
    }

person.fullName = "Jane Smith"
```


Member functions

- Classes can also contain functions
- Declare functions as shown in *Functions* in Lesson 2
  - `fun` keyword
  - Can have default or required parameters
  - Specify return type (if not `Unit`)

Inheritance

Inheritance

- Kotlin has single-parent class inheritance
- Each class has exactly one parent class, called a superclass
- Each subclass inherits all members of its superclass including ones that the superclass itself has inherited

If you don't want to be limited by only inheriting a single class, you can define an interface since you can implement as many of those as you want.

Interfaces

- Provide a contract all implementing classes must adhere to
- Can contain method signatures and property names
- Can derive from other interfaces

Format: `interface NameOfInterface { interfaceBody }`

Interface example

```
interface Shape {
    fun computeArea() : Double
}

class Circle(val radius:Double) : Shape {
    override fun computeArea() = Math.PI * radius * radius
}

val c = Circle(3.0)
println(c.computeArea())
=> 28.274333882308138
```


Extending classes

To extend a class:

- Create a new class that uses an existing class as its core (subclass)
- Add functionality to a class without creating a new one (extension functions)

Creating a new class

- Kotlin classes by default are not subclassable
- Use `open` keyword to allow subclassing
- Properties and functions are redefined with the `override` keyword

Classes are final by default

Declare a class

```
class A
```

Try to subclass A

```
class B : A

=>Error: A is final and cannot be inherited from
```


Use open keyword

Use `open` to declare a class so that it can be subclassed.

Declare a class

```
open class C
```

Subclass from C

```
class D : C()
```


Overriding

- Must use `open` for properties and methods that can be overridden (otherwise you get compiler error)
- Must use `override` when overriding properties and methods
- Something marked `override` can be overridden in subclasses (unless marked `final`)

Abstract classes

- Class is marked as `abstract`
- Cannot be instantiated, must be subclassed
- Similar to an interface with the added the ability to store state
- Properties and functions marked with `abstract` must be overridden
- Can include non-abstract properties and functions

Example abstract classes

```
abstract class Food {
  abstract val kcal : Int
  abstract val name : String
  fun consume() = println("I'm eating ${name}")
}
class Pizza() : Food() {
  override val kcal = 600
  override val name = "Pizza"
}
fun main() {
  Pizza().consume()   // "I'm eating Pizza"
}
```


When to use each

- Defining a broad spectrum of behavior or types? Consider an interface.
- Will the behavior be specific to that type? Consider a class.
- Need to inherit from multiple classes? Consider refactoring code to see if some behavior can be isolated into an interface.
- Want to leave some properties / methods abstract to be defined by subclasses? Consider an abstract class.
- You can extend only one class, but implement one or more interfaces.

Extension functions

Extension functions

Add functions to an existing class that you cannot modify directly.

- Appears as if the implementer added it
- Not actually modifying the existing class
- Cannot access private instance variables

Format: `fun ClassName.functionName( params ) { body }`

Why use extension functions?

- Add functionality to classes that are not open
- Add functionality to classes you don't own
- Separate out core API from helper methods for classes you own

Define extension functions in an easily discoverable place such as in the same file as the class, or a well-named function.

Extension function example

Add `isOdd()` to `Int` class:

```
fun Int.isOdd(): Boolean { return this % 2 == 1 }
```

Call `isOdd()` on an `Int`:

```
3.isOdd()
```

Extension functions are very powerful in Kotlin!

Special classes

Data class

- Special class that exists just to store a set of data
- Mark the class with the `data` keyword
- Generates getters for each property (and setters for vars too)
- Generates `toString()`, `equals()`, `hashCode()`, `copy()` methods, and destructuring operators

Format: `data class <NameOfClass>( parameterList )`

Data class example

Define the data class:

```
data class Player(val name: String, val score: Int)
```

Use the data class:

```
val firstPlayer = Player("Lauren", 10)
println(firstPlayer)
=> Player(name=Lauren, score=10)
```

Data classes make your code much more concise!

Pair and Triple

- `Pair` and `Triple` are predefined data classes that store 2 or 3 pieces of data respectively
- Access variables with `.first`, `.second`, `.third` respectively
- Usually named `data` classes are a better option (more meaningful names for your use case)

Pair and Triple examples

```
val bookAuthor = Pair("Harry Potter", "J.K. Rowling")
println(bookAuthor)
=> (Harry Potter, J.K. Rowling)

val bookAuthorYear = Triple("Harry Potter", "J.K. Rowling", 1997)
println(bookAuthorYear)
println(bookAuthorYear.third)
=> (Harry Potter, J.K. Rowling, 1997)
   1997
```

Pair to

`Pair`'s special `to` variant lets you omit parentheses and periods (infix function).

It allows for more readable code

```
val bookAuth1 = "Harry Potter".to("J. K. Rowling")
val bookAuth2 = "Harry Potter" to "J. K. Rowling"
=> bookAuth1 and bookAuth2 are Pair (Harry Potter, J. K. Rowling)
```

Also used in collections like Map and HashMap

```
val map = mapOf(1 to "x", 2 to "y", 3 to "zz")
=> map of Int to String {1=x, 2=y, 3=zz}
```

Enum class

User-defined data type for a set of named values

- Use `this` to require instances be one of several constant values
- The constant value is, by default, not visible to you
- Use `enum` before the `class` keyword

Format: `enum class EnumName { NAME1, NAME2, … NAMEn }`

Referenced via `EnumName.<ConstantName>`

Enum class example

Define an `enum` with red, green, and blue colors.

```
enum class Color(val r: Int, val g: Int, val b: Int) {
    RED(255, 0, 0), GREEN(0, 255, 0), BLUE(0, 0, 255)
}

println("" + Color.RED.r + " " + Color.RED.g + " " + Color.RED.b)
=> 255 0 0
```

Object/singleton

- Sometimes you only want one instance of a class to ever exist
- Use the `object` keyword instead of the `class` keyword
- Accessed with `NameOfObject.<function or variable>`

Object/singleton example

```
object Calculator {
    fun add(n1: Int, n2: Int): Int {
        return n1 + n2
    }
}

println(Calculator.add(2,4))
=> 6
```

Companion objects

- Lets all instances of a class share a single instance of a set of variables or functions
- Use `companion` keyword
- Referenced via `ClassName.PropertyOrFunction`

Companion object example

```
class PhysicsSystem {
    companion object WorldConstants {
        val gravity = 9.8
        val unit = "metric"
        fun computeForce(mass: Double, accel: Double): Double {
            return mass * accel
        }
    }
}
println(PhysicsSystem.WorldConstants.gravity)
println(PhysicsSystem.WorldConstants.computeForce(10.0, 10.0))

=> 9.8100.0
```

Organizing your code

Single file, multiple entities

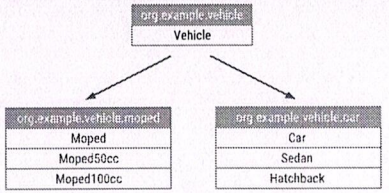
- Kotlin DOES NOT enforce a single entity (class/interface) per file convention
- You can and should group related structures in the same file
- Be mindful of file length and clutter

Packages

- Provide means for organization
- Identifiers are generally lower case words separated by periods
- Declared in the first non-comment line of code in a file following the `package` keyword

```
package org.example.game
```


Example class hierarchy



Visibility modifiers

Use visibility modifiers to limit what information you expose.

- `public` means visible outside the class. Everything is public by default, including variables and methods of the class.
- `private` means it will only be visible in that class (or source file if you are working with functions).
- `protected` is the same as `private`, but it will also be visible to any subclasses.

Summary

Summary

In Lesson 3, you learned about:

- Classes, constructors, and getters and setters
- Inheritance, interfaces, and how to extend classes
- Extension functions
- Special classes: data classes, enums, object/singletons, companion objects
- Packages
- Visibility modifiers

Pathway

Practice what you've learned by completing the pathway:

Lesson 3: Classes and objects