**PROJEKTI**

**Hyrje rreth detyrës**

Në këtë projekt ne jemi munduar në mënyrë sa më të qartë ta shpjegojmë kodin sipas të cilit është zhvilluar aplikacioni jonë, duke filluar nga mënyra se si është zgjidhur, pjesët kryesore të kodit, arkitektura e tij etj. Na është dhënë një shembull i një “Pie Chart” që është gjeneruar nga një “output-view object”. Sipas tabelës së specifikimit në të cilën është dhënë klasa PieChartWriter kemi zhvilluar apo formuluar këtë klasë të cilën më pas:

1. E kemi testuar me një aplikacion tjetër prej nga është bërë edhe paraqitja e Pie Chart “How I spend my day”.
2. E kemi modifikuar që të zhvillojmë një aplikacion që kur ekzekutohet shfaq **të hyrat** dhe **të dalat** e qeverisë së SHBA-ve në vitin Fiskal 1997, natyrisht në bazë të të dhenave paraprake që janë japur (*INCOME and OUTLAYS*). Edhe kësaj i është bashkangjitur një klasë testuese që është e nevojshme për paraqitjen grafike.

Thene me shkurt detyra ka kerkuar nga ne krijimin e nje grafi te formes “pie” i cili funksionon duke marrur parametrat nga nje klase tjeter.

**Si është zgjidhur problemi**

Ne pergjithesi problem edhe zgjidhur duke krijuar metoda me parametra te caktuara te cilat keto parametra i vendosin si vlera ne ndryshore te fushes te krijuara me pare dhe pastaj e me ane te metodes **repaint()** krijojne arc apo sektore te rrethit me ato vlera .

Me hollesisht brenda klasës PieChartWriter së pari kemi deklaruar variablat e fushës përkatëse gjithmonë duke u bazuar në tabelën e specifikimit: **String sentence1=""; String sentence2="";**--- e kështu me radhë deri te **String sentence6="";**; String variablat janë inicializuar me stringje bosh **""**. E njëjta vlen edhe për **Color color1;---** vetëm se këtu nuk janë inicializuar fare. Variablave **int amount1 = 0;**, që janë variabla të tipi int iu është dhene vlera 0 që të gjithave, ashtu që të mund të ndryshojmë më vonë sasinë apo sipërfaqën e psh.**Slice1** në bazë të vlerave që do tju shtojmë. E njejta eshte zbatuar edhe te int **vlera1 = 0**,int **vlera2 = 0**--- deri te **vlera6**.

Pastaj kemi metodat që janë dhënë në tabelën e specifikimit. Së pari kemi metodën:**setSlice1(String label, int amount, Color c)**, ku brenda kësaj metode e barazojmë vlerën **vlera1** më metodën **Math.round** dhe ja shtojmë para metodës **(int)** ashtu që rezultati të jetë i tipit int.Pra kemi: **vlera1= (int) Math.round(amount\*15);**, Në mënyrë të ngjashme e barazojmë **sentence1 = label;** pra vlerën e variablës së fushës e ngarkojme me label-parametrin e metodës **setSlice1**, dhe në fund **color1 = c;**.

Në mënyrë të ngjashme janë shkruar edhe metodat tjera me radhë:**setSlice2---setSlice6**.

Në vazhdim kemi shkruar metodën **public void paintComponent(Graphics g)**, që referencohet apo thirret me metoda tjera përkatëse .Pra është metodë shumë e rëndësishme dhe e përdorshme; është metoda në të cilën realizohet vizatimi i caktuar. Brenda saj kemi:

* Metoden **g.setColor(color1);**- që merr ngjyrën **color1** nga **setSlice1**
* Metodën **g.fillArc(80,80,300,300,0,vlera1); -** ku katër parametrat e parë me vlerat e dhëna përcaktojnë pozitën e rrethit, ndërsa dy të fundit hapësirën që zë **Slice1** në rreth. Edhe kjo merr **vlera1** nga setSlice1
* Metodën **g.drawString( sentence1,440,200);** - që merr sentence poashtu nga setSlice1 dhe dy parametrat e fundit që përcaktojnë pozitën e fjalise në panel.

Kjo metodë vepron ngjashëm edhe me metodat tjera **setSlice**, natyrisht vlera e **0** zëvendësohet me **amount1** dhe **vlera1** me **vlera2**. Gjithashtu ndryshojnë kordinatat e pozitave të fjalive.

Përveq metodave të japura në tabelën e specifikimit, duke u bazuar në klasën testuese TestPieChart , vërejmë se është thirrur edhe një metodë setTitle. Prandaj në klasën PieChartWriter formojmë metodën : **setTitle**, brenda kësaj metode kemi krijuar kornizën e re me titull dhe përmasat që i kemi caktuar. Kjo kornizëështë e dukshme dhe mbyllet automatikisht kur merr mesazh nga sistemi operativ.

Ekzekutimi i kësaj klase (PieChartWriter) bëhet me anë të klasës tjetër testuese që ka metodë kryesore dhe me krijimin e një objekti të ri: **PieChartWriter p = new PieChartWriter();** , të klasës paraprake dhe me anë të variablës **p** i ka thirrë metodat e klasës përkatëse:

**p.setTitle("How I spend my day");**

**p.setSlice1("Sleep: 7 hours", 7, Color.black);**

**p.setSlice4("Recreation: 9 hours", 9, Color.gray);**

Pra klasa testuese për ti thirrë këto metoda duhet që lloji i parametrave të jetë i njëjtë. Poashtu mund të mos e thirrim ndonjë metodë dhe renditja e metodave këtu nuk luan rëndësi. Qëllimi kryesor i klasave testuese është që nuk kemi nevojë të mbajmë në mend brendësinë apo trupin e metodave, mjafton që ti shkruajmë njëherë brenda një klase dhe mund ta përdorim shumë herë atë klase në këtë rast PieChartWriter, me klasa të ndryeshme testuese siq është edhe në rastin tonë.

**Pjesët kryesore të kodit**

Pjesët kryesore të këtij kodi janë:

* metodat **setSlice** për të cilat folëm edhe më herët. Me anë të tyre i ndajme Slices apo pjesët përkatëse brenda rrethit:

**public void setSlice1(String label, int amount, Color c)**

**//**ballina e metodes tregon se cilet parametra ipen kur thirret metoda

**{ vlera1= (int) Math.round(amount\*15)**

**//**caktimi i vlere se harkut te pare duke marre parametrin dhe shumezuar me harku minimal te mundshem ne baze te vlerave te dhena ,ne qofte se harku minimal eshte i tipit double si ne rastin e dyte ateher **Math.round** e rrumbullaksen duke e shendrruar ne te dhene te tipit long dhe pastaj me ane te fjales kyqe **(int)** e shendrron ne te dhene te tipit integer.

**amount1=vlera1;**

**//**ruhet vlera e kendeve te vizatuara deri me tani ne menyre qe te dihet se ku do te filloj vizatimi i kendit te radhes.

**sentence1 = label; //**i caktohet stringu nga parametrat

**color1 = c; //** i caktohet ngjyra e marre nga parametrat

**repaint(); //** ben thirrje e metodes paint component **}**

* metoda **paintComponent** që bën vizatimin përkatës:

**public void paintComponent(Graphics g )**

**{ g.setColor(color1); //** cakton ngjyren qe do te perdoret

**g.fillArc(80,80,300,300,0,vlera1);** // dy vlerat e fundit caktojn kendin ku fillon te vizatohet harku dhe madhesin apo kendin qe do ta mbuloje

**g.drawString( sentence1,170,20);**

**….**

**g.setColor(color6);**

**g.fillArc(80,80,300,300,amount5,vlera6);**

**g.drawString( sentence6,440,300);}**

**}**

* metoda **SetTitle(String title)** ben krijimin e kornizes dhe vendosje e vizatimeve ne te duke ia cakrtuar edhe emrin kornizes:

**JFrame f = new JFrame() //**krijon nje instance te JFrame pra kornizes duke e ia caktuar atij objekti emrin f

**f.getContentPane().add(this);//**i jep urdher per te marre vizatimet dhe vendosur ne kornize **f.setTitle(title);//** i vendos emrin kornizes **f.setSize(700,500);//**cakton madhesine e kornizes **f.setVisible(true);//** e ben kornizen te dukshme **f.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);//** ben qe te perfundoj program me rastine mbylljes se kornizes

Rëndësi të veqantë kanë edhe edhe paketat **java.awt,\*;** dhe**javax.swing.\*;** që mundësojnë kryerjen e këtyre vizatimeve me anë të klasëve që përmbajnë dhe që janë në dispozicion të përdoruesit kur shkruajmë psh **import java.awt.\*;**.

Për rastin apo detyrën 2 vlejnë të gjitha këto që diskutuam deri tani me disa ndryshime të vogla, ku te vlera1 në vend të**vlera1 = (int) Math.round(amount\*15);** kemi **vlera1 = (int) Math.round(amount\*(360.0/100));**. E njëjta vlen edhe për **vlera2** e **Slice2** e kështu me radhë deri te **Slice6**.

Ekzekutimi i këtij rasti natyrisht që ka klasë testuese tjetër me metodë kryesore që e kemi emëruar TestPieChart1. Në këtë klasë i kemi krijuar 2 objekte të reja nga klasa PieChartWriter, pasi që duhet të na shfaqen 2 rrathë me përqindje të Slice të ndryshme(INCOME and OUTLAYS), prandaj kemi:

**PieChartWriter p = new PieChartWriter();**

**p.setTitle("INCOME:");**

**p.setSlice1("personal income taxes: 46%", 46, Color.black);**

**p.setSlice5("social security and medicare taxes: 34%", 34, Color.gray);**

**….setSlice2, setSlice3, setSlice4**

**PieChartWriter a= new PieChartWriter();**

**a.setTitle("OUTLAYS:");**

**a.setSlice1("social security and medicare: 38%", 38, Color.black);**

**a.setSlice6("general government: 2%", 2, Color.orange);**

**…. setSlice2, setSlice3, setSlice4, setSlice5**

Edhe këtu numri i parametrave është i njëjtë, gjithashtu mund të mos e thirrim ndonjë metodë dhe renditja e metodave këtu nuk luan rëndësi.

**Arkitektura**

**KlasaTestPieChart C**

**main**

**KlasaPieChartWriter M**

**~sentence1 String …**

**~color1 Color …**

**~amount1 int ….**

**~vlera 1 int….**

**+setTitle**

**+setSlice1…**

**+paintComponent**

**Klasa Math**

**Math.round()**

**JPanel V**

**Si do te zhvillonit edhe njehere prej fillimi te njejten detyre?**

Ne qofte se do duhej te krijohej nje Pie Chart i cili do te shfaqte grafikun me informacionet e dhena do te ishte duke e perdorur klasen e gatshme PieChart te libraries Java API(Application Programming Interface) e cila na mundeson te shfaqim grafikun duke kerkuar nga ne qe vetem te modifikojme informacionet te cilat ne ia japim per ti shfaqur.

Ne qofte se nuk do te perdornim Java api ateher do te benim ndryshimin e emertimit te parametrava te ngjyres , stringut dhe int duke mos pasur nevoj te deklarojme variabla te reja per secilen nga metodat.Gjithashtu do te krijonim nje variable te re e cila do te mblidhte shumen e te gjitha vlerave numerike int te dhene si parametra dhe do njehesonte kendin qe do te paraqiste nje njesi elementare e te dhenave psh. 1 ore ose 1% ne menyre qe te mos kete nevoje per nderhyrje ne kod sa here qe te nderrohen parametrat dhe te dy klaset testPieCharter dhe testPieCharter 2 do te mund te funksionin pa ia nderruar vlerat(360/24) me (360/100). Kjo ndryshore do te ishte e tipit double ne menyre qe te ruhet sa me shume saktesia ne njehesim.