

2025

# ATOLL PROJEKAT

PRVA FAZA

STEFAN CVETKOVIĆ 19461 | LAZAR STANKOVIĆ 19375

## Contents

IZGLED IGRE.....	3
Početni prozor .....	3
Tabla za igranje .....	4
Light mode .....	5
Odabrana/Odigrana polja.....	6
FUNKCIJE IGRE .....	7
main.py .....	7
options.py.....	7
openView() .....	7
startButtonClick() .....	10
exitButtonClick() .....	11
swapPlayerClick() .....	11
addPlayerClick() .....	11
changeTheme() .....	11
board.py .....	12
openView(m, isbot, playerfirst, mainapp, currentTheme) .....	12
startPage() .....	13
infoFrameFill(frame) .....	14
confirmClick().....	15
exitClick() .....	16
undoClick().....	16
getCurrentPlayerName().....	16
getCurrentPlayer() .....	17
render_board(parent, matrix, rows, cols) .....	17
create_button(parent, tile, row, col) .....	18
on_tile_click(r, c) .....	19
changeAppTheme .....	19
enums.py.....	20
eTile .....	20
tableCreator.py .....	20

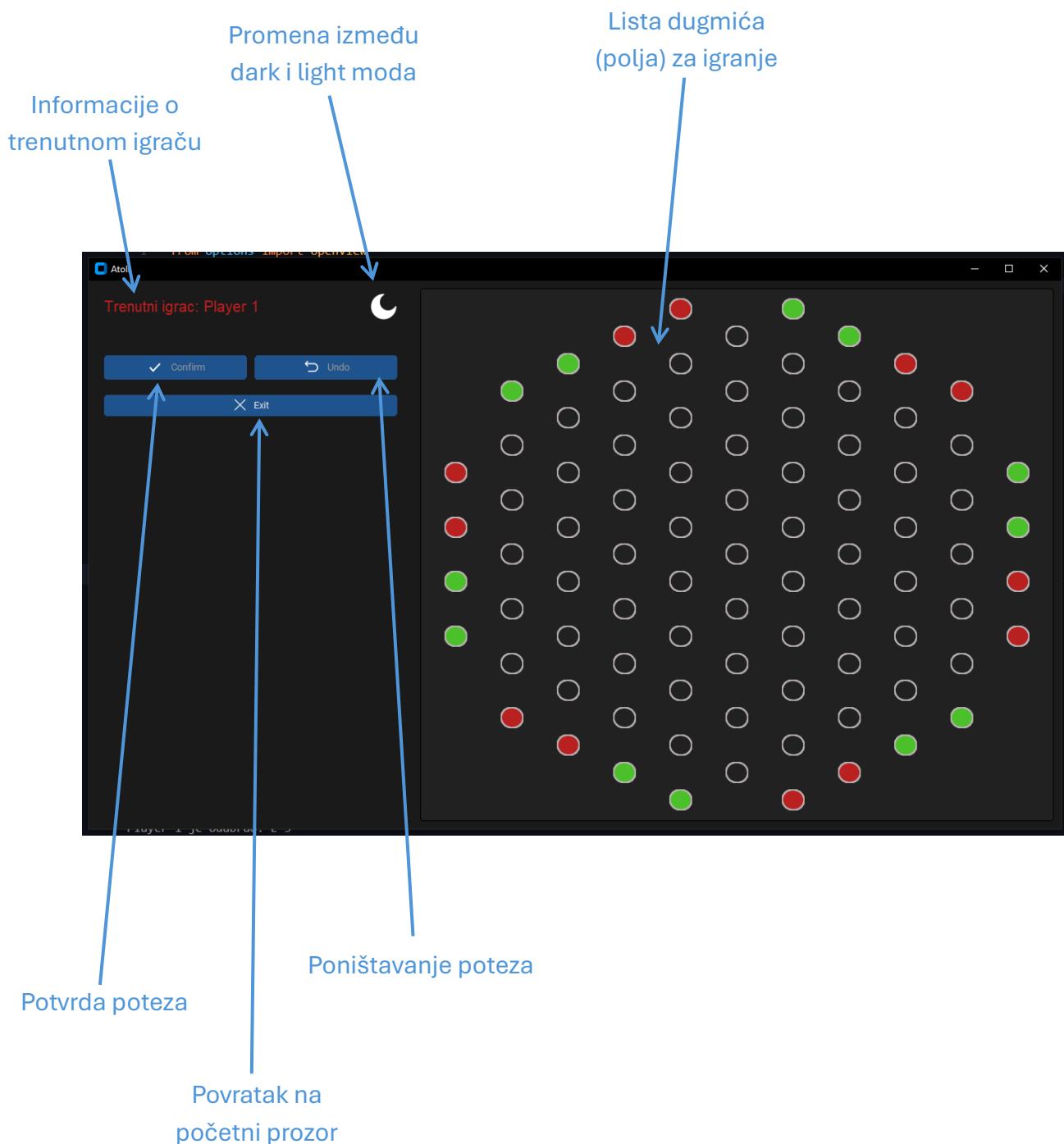
createTable(n) .....	20
createEmptyTable(matrixN, matrixM).....	21
fillTable(matrix, n, matrixN) .....	21
fillEdgeValues(matrix, n, matrixN).....	22
fillMiddleValues(matrix, n) .....	23
fillSeparator(matrix, n, m) .....	23
INFORMACIJE O IGRI .....	24
Tehnologije.....	24
Reference .....	24

## IZGLED IGRE

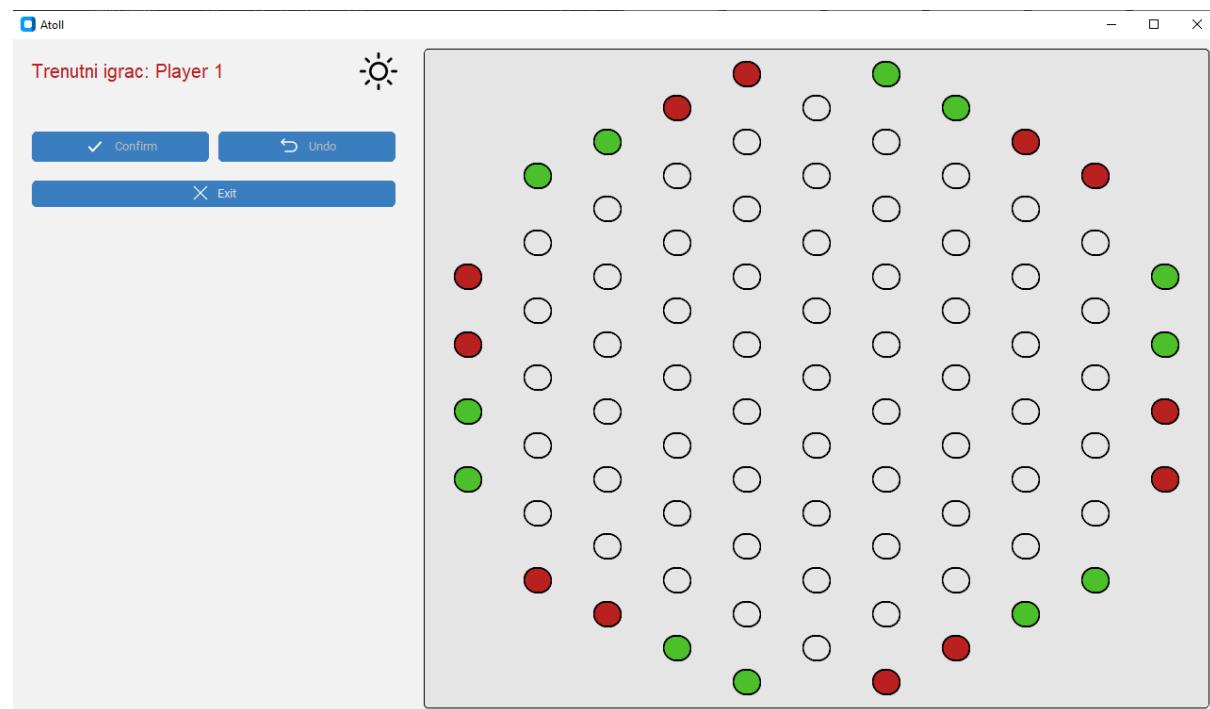
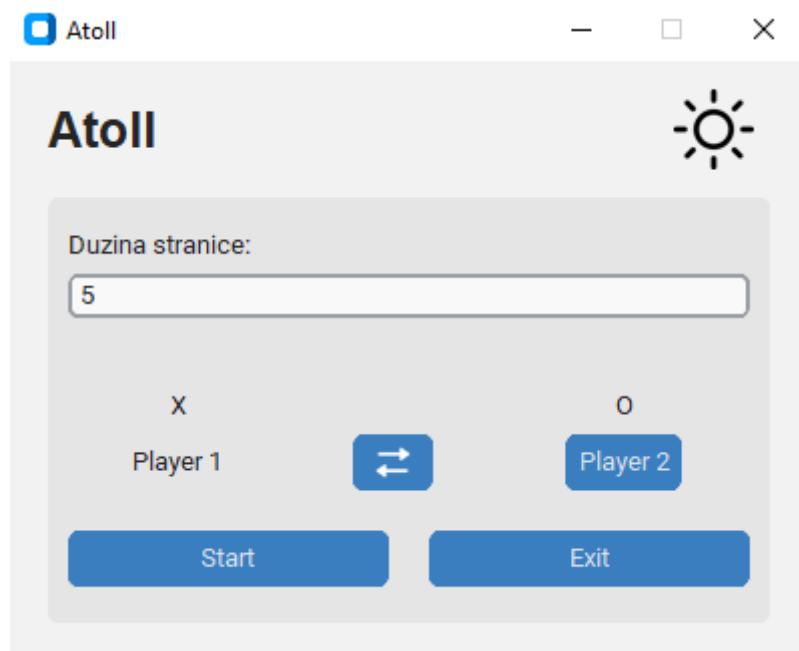
## Početni prozor



## Tabla za igranje

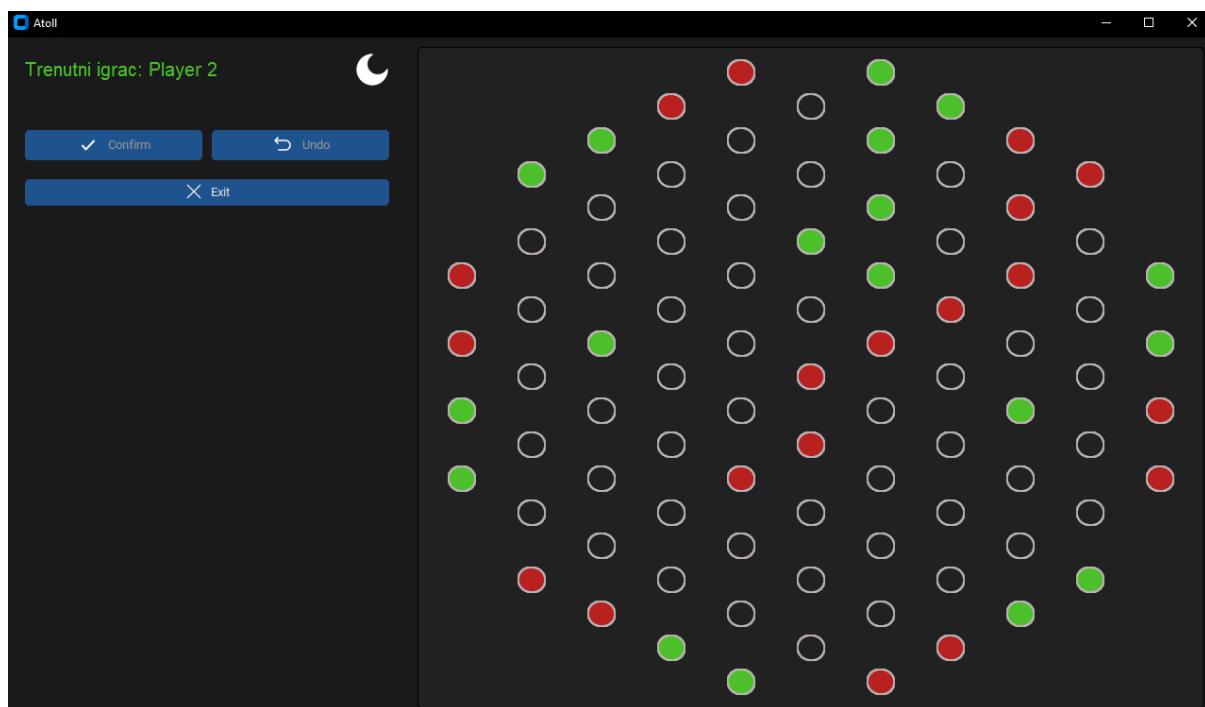


## Light mode



## Odabrana/Odigrana polja

```
Player 1 je odabrao: G 3
Player 1 je odabrao: G 4
Player 1 je odabrao: H 5
Player 1 je odabrao: G 4
Player 1 je odabrao: G 5
Player 1 je odabrao: H 4
Player 1 je odigrao: H 4
Player 2 je odabrao: F 2
Player 2 je odigrao: F 2
Player 1 je odabrao: H 5
Player 1 je odigrao: H 5
Player 2 je odabrao: F 3
Player 2 je odigrao: F 3
Player 1 je odabrao: G 5
Player 1 je odigrao: G 5
Player 2 je odabrao: E 3
Player 2 je odigrao: E 3
```



## FUNKCIJE IGRE

### main.py

```
from options import openView

openView()
```

Koristi se za pokretanje aplikacije i poziva funkciju iz options.py što je ujedno i početna stranica.

### options.py

#### openView()

```
def openView():
    global app, nTextEntry, labelP, botButton, swapped, isBot
    customtkinter.set_appearance_mode('dark')
    customtkinter.set_default_color_theme('dark-blue')

    swapped = True
    isBot = True

    app = customtkinter.CTk()
    app.title('Atoll')
    app.geometry('400x300')
    app.minsize(400, 300)
    app.maxsize(400, 300)
    app.resizable(False, False)
    app.grid_columnconfigure(0, weight=1)
    app.grid_rowconfigure(1, weight=1)

    header = customtkinter.CTkFrame(app, fg_color="transparent")
    header.grid(row=0, column=0, columnspan=2, sticky="ew", padx=10)
    header.grid_columnconfigure(0, weight=1)
    header.grid_columnconfigure(1, weight=0)

    gameNameLabel = customtkinter.CTkLabel(header, text='Atoll',
    fg_color='transparent', font=(' ', 25, 'bold'))
    gameNameLabel.grid(row=0, column=0, padx=10, pady=10, sticky="w")

    theme_icon = customtkinter.CTkImage(
        light_image=Image.open("./assets/light_theme.png"),
```

```
dark_image=Image.open("./assets/dark_theme.png"),
size=(40, 40)
)

button = customtkinter.CTkButton(
    header,
    text="",
    width=40,
    height=40,
    hover=None,
    fg_color="transparent",
    image=theme_icon,
    command=changeTheme
)
button.grid(row=0, column=1, padx=10, pady=10, sticky="e")

frame = customtkinter.CTkFrame(app)
frame.grid(row=1, column=0, padx=20, pady=(0, 20), sticky='nws')
frame.grid_columnconfigure(0, weight=1)

nInputFrame = customtkinter.CTkFrame(frame, fg_color='transparent')
nInputFrame.grid(row = 0, column = 0, padx=10, pady=(10, 20), sticky='ew',
columnspan=2)
nInputFrame.grid_columnconfigure(0, weight=1)

nInputLabel = customtkinter.CTkLabel(nInputFrame, text='Duzina stranice:')
nInputLabel.grid(row = 0, column = 0, sticky='w')

nTextEntry = customtkinter.CTkEntry(nInputFrame, height=10,
placeholder_text='5, 7, 9...')
nTextEntry.grid(row = 1, column = 0, sticky='we')

playerFrame = customtkinter.CTkFrame(frame, fg_color='transparent')
playerFrame.grid(row = 1, column = 0, padx = 10, pady = 10, sticky = 'ew',
columnspan=2)
playerFrame.grid_columnconfigure((0, 1, 2), weight=1)

labelX = customtkinter.CTkLabel(playerFrame, text='X')
labelX.grid(row = 0, column = 0)

labelO = customtkinter.CTkLabel(playerFrame, text='O')
labelO.grid(row = 0, column = 2)

labelP = customtkinter.CTkLabel(playerFrame, text='Player 1', width=40)
labelP.grid(row = 1, column = 0, padx = 10)
```

```
swapImage = customtkinter.CTkImage(Image.open('./assets/swap.png'),  
size=(20, 20))  
  
swapButton = customtkinter.CTkButton(playerFrame, text='',  
image=swapImage, width=40, command=swapPlayerClick)  
swapButton.grid(row=1, column=1, padx = 10)  
  
botButton = customtkinter.CTkButton(playerFrame, text='Bot', width=40,  
command=addPlayerClick)  
botButton.grid(row=1, column=2, padx = 10)  
  
buttonsFrame = customtkinter.CTkFrame(frame, fg_color='transparent')  
buttonsFrame.grid(row = 2, column=0, sticky='ew', columnspan=2)  
buttonsFrame.grid_rowconfigure(0, weight=1)  
buttonsFrame.grid_columnconfigure(0, weight=1)  
buttonsFrame.grid_columnconfigure(1, weight=1)  
  
startButton = customtkinter.CTkButton(buttonsFrame, text='Start',  
command=startButtonClick)  
startButton.grid(row=0, column=0, padx=10, pady=10, sticky='ew')  
  
exitButton = customtkinter.CTkButton(buttonsFrame, text='Exit',  
command=exitButtonClick)  
exitButton.grid(row=0, column=1, padx=10, pady=10, sticky='ew')  
app.mainloop()
```

Funkcija koju poziva main.py kako bi kreirao sam prozor i započeo aplikaciju. Funkcija stavlja inicijalne vrednosti za globalne promenljive i kreira sam izgled stranice. Podešava se izgled prozora aplikacije i zabranjuje se *resize*.

### startButtonClick()

```

def startButtonClick():
    notNum = False
    n = nTextEntry.get()

    try:
        n = int(n)
    except:
        notNum = True

    if(notNum or n < 3 or n > 9 or n % 2 == 0 or isBot):
        infoBox = customtkinter.CTkToplevel()
        infoBox.geometry('300x100')
        infoBox.grid_rowconfigure(0, weight=1)
        infoBox.grid_columnconfigure(0, weight=1)
        infoBox.title('Notification')

        label = customtkinter.CTkLabel(infoBox, text='N mora biti neparan broj
izmedju 3 i 9')

        if(isBot):
            label.configure(text = 'Igranje protiv racunara nije omoguceno')

        label.grid(row = 0, column = 0, sticky='nsew')
        infoBox.grab_set()
        return

    app.withdraw()
    openGame(n, isBot, swapped, app,
customtkinter.get_appearance_mode().lower())

```

Funkcija koja se poziva klikom na start dugme. U ovoj funkciji se vrši validacija ulaznih parametra i prikazuje odgovarajući prozor u zavisnosti njihove validnosti. Ukoliko se pokrene igra protiv računara dobijamo poruku da je igranje protiv računara onemogućeno. Ukoliko je za veličinu stranice uneta vrednost koja nije paran broj ili vrednost koja nije između 3 i 9 korisniku se prikaže poruka koja ga obaveštava da vrednost mora biti neparan broj između 3 i 9. U slučaju da je sve ispravno trenutni prozor se sakrije i učita se tabla pozivom funkcije openGame().

### exitButtonClick()

```
def exitButtonClick():
    app.quit()
```

Funkcija koja se koristi za prekidanje rada igre.

### swapPlayerClick()

```
def swapPlayerClick():
    global swapped
    swapped = not swapped
    labelP.grid(row = 1, column = 0 if swapped else 2)
    botButton.grid(row = 1, column = 2 if swapped else 0)
```

Funkcija koja se koristi za odabir igrača koji će igrati prvi.

### addPlayerClick()

```
def addPlayerClick():
    global isBot
    isBot = not isBot
    botButton.configure(text = 'Bot' if isBot else 'Player 2')
```

Funkcija koja vrši promenu između igranja protiv računara i drugog igrača.

### changeTheme()

```
def changeTheme():
    customtkinter.set_appearance_mode("dark" if
customtkinter.get_appearance_mode().lower() == "light" else "light")
```

Funkcija koja vrši promenu između *light* i *dark* teme.

## board.py

```
openView(m, isbot, playerfirst, mainapp, currentTheme)
```

```
def openView(m, isbot, playerfirst, mainapp, currentTheme):
    global n, isBot, buttons, matrix, player1First, mainApp, lastMove, app,
players, playerColors, currentPlayer
    n = int(m)
    isBot = isbot
    buttons = []
    matrix = []
    player1First = playerfirst
    lastMove = None
    mainApp = mainapp
    currentPlayer = 0 if player1First else 1
    players = (eTile.Player1.value[0], eTile.Player2.value[0])
    playerColors = ('#B92020', '#4BC02A')

    customtkinter.set_appearance_mode(currentTheme)
    customtkinter.set_default_color_theme('dark-blue')

    app = customtkinter.CTkToplevel()
    app.protocol("WM_DELETE_WINDOW", exitClick)
    app.geometry("1280x720")
    app.title("Atoll")
    app.grid_columnconfigure(0, weight=2)
    app.grid_columnconfigure(1, weight=10)
    app.grid_rowconfigure(0, weight=1)

    startPage()
    app.mainloop()
```

Ova funkcija kreira tablu na kojoj se igra. Ona podešava globalne promenljive koje će se koristiti za rad aplikacije i podešava izgled. Pored toga ona podešava veličinu samog prozora i pokreće glavni loop igre.

Kao parametre prima: *m* - uneta dužina sa *options.py* pogleda, *isBot* – informaciju da li je igra protiv računara ili čoveka, *playerfirst* – informaciju da li je „Player 1“ prvi igrač, *mainapp* – referenca na *options.py* pogled, *currentTheme* – poslednju temu koja je bila odabrana na *options.py* pogledu.

## startPage()

```
def startPage():
    global buttons
    global matrix
    matrix, matrixN, matrixM = createTable(n)

    infoFrame = customtkinter.CTkFrame(app, fg_color="transparent")
    infoFrame.grid(row = 0, column = 0, sticky='nwse')
    infoFrame.grid_columnconfigure(0, weight=1)

    infoFrameFill(infoFrame)

    board_frame = customtkinter.CTkFrame(app, border_color='#000',
    border_width=1)
    board_frame.grid(row = 0, column = 1, sticky='nwse', padx=10, pady=10)
    board_frame.grid_rowconfigure(0, weight=1)
    board_frame.grid_columnconfigure(0, weight=1)

    boardHolder = customtkinter.CTkFrame(board_frame, fg_color='transparent')
    boardHolder.grid(row = 0, column = 0, padx = 10, pady = 10, sticky='nswe')

    for i in range(matrixN):
        boardHolder.grid_rowconfigure(i, weight=1)

    for i in range(matrixM):
        boardHolder.grid_columnconfigure(i, weight=1)

    buttons = render_board(boardHolder, matrix, matrixN, matrixM)
```

Ova funkcija kreira matricu koja se za samu tabelu polja u igri, pozivajući funkciju `createTable(n)` iz fajla `tableCreator.py`. Ona kreira pogled koji se deli na side panel (informacije o trenutnom igraču i akcioni dugmići) i tablu sa poljima. Side panel se popunjava pozivom funkcije `infoFrameFill(infoFrame)` dok se tabela popunjava funkcijom `render_board(boardHolder, matrix, matrixN, matrixM)`.

## infoFrameFill(frame)

```

def infoFrameFill(frame):
    global confirmButton, undoButton, currentPlayerLabel

    helperFrame = customtkinter.CTkFrame(frame, fg_color="transparent")
    helperFrame.grid(row = 0, column = 0, sticky='ew')
    helperFrame.grid_rowconfigure(0, weight=1)
    helperFrame.grid_columnconfigure(0, weight=1)

    currentPlayerLabel = customtkinter.CTkLabel(helperFrame,
text=getCurrentPlayer(), font=(' ', 20), text_color=
playerColors[currentPlayer])
    currentPlayerLabel.grid(row = 0, column = 0, padx = 20, pady = 20,
sticky='w')

    theme_icon = customtkinter.CTkImage(
        light_image=Image.open("./assets/light_theme.png"),
        dark_image=Image.open("./assets/dark_theme.png"),
        size=(40, 40)
    )

    themeButton = customtkinter.CTkButton(
        helperFrame,
        text="",
        width=40,
        height=40,
        hover=None,
        fg_color='transparent',
        border_width=0,
        image=theme_icon,
        command=changeAppTheme
    )
    themeButton.grid(row=0, column=1, padx=10, pady=10, sticky='nwse')

    buttonsFrame = customtkinter.CTkFrame(frame, fg_color='transparent')
    buttonsFrame.grid(row = 1, column = 0, padx = 20, pady = 20, sticky='we')
    buttonsFrame.grid_columnconfigure(0, weight=1)
    buttonsFrame.grid_columnconfigure(1, weight=1)
    buttonsFrame.grid_rowconfigure(0, weight=1)

    confirmIcon = Image.open('./assets/checkLight.png')
    undoIcon = Image.open('./assets/undoLight.png')
    cancelIcon = Image.open('./assets/cancelLight.png')

    confirmButton = customtkinter.CTkButton(buttonsFrame, text='Confirm',
state='disabled', command=confirmClick,
image=customtkinter.CTkImage(light_image=confirmIcon, size=(24, 24)))

```

```

    confirmButton.grid(row = 0, column = 0, sticky='nswe', pady=10, padx=(0, 5))

    undoButton = customtkinter.CTkButton(buttonsFrame, text='Undo',
state='disabled', command=undoClick,
image=customtkinter.CTkImage(light_image=undoIcon, size=(24, 24)))
    undoButton.grid(row = 0, column = 1, sticky='nswe', pady=10, padx=(5, 0))

    exitButton = customtkinter.CTkButton(buttonsFrame, text='Exit',
command=exitClick, image=customtkinter.CTkImage(light_image=cancelIcon,
size=(18, 18)))
    exitButton.grid(row = 1, column = 0, sticky='we', columnspan = 2, pady=10)

```

Ova funkcija kreira sve potrebne elemente koji se nalaze u side panelu. To su dugme za potvrdu i poništenje poteza, dugme za izlaz iz igre i dugme za promenu između dark i light moda. Pored toga ova funkcija takođe dodaje label koji nam govori koji igrač je na potezu uz tekst koji prati boju trenutnog igrača.

### confirmClick()

```

def confirmClick():
    global currentPlayerLabel, lastMove, currentPlayer, n
    print(f"{getCurrentPlayerName()} je odigrao: {chr(ord('A') - 1 +
lastMove[1])} {((lastMove[0] + lastMove[1] - n + 1)//2)}")

    currentPlayer = (currentPlayer + 1) % 2

    confirmButton.configure(state = 'disabled')
    undoButton.configure(state = 'disabled')
    lastMove = None

    currentPlayerLabel.configure(text = getCurrentPlayer(), text_color=
playerColors[currentPlayer])

```

Funkcija koju poziva Confirm dugme. Ova funkcija menja trenutnog igrača i samim tim isključuje confirm i undo dugme. Ovi dugmići moraju biti isključeni pri promeni jer sledeći igrač nije odabrao svoj potez. Pored toga ova funkcija takođe štampa u konzoli koje polje je odigrano.

### exitClick()

```
def exitClick():
    app.withdraw()
    app.quit()
    mainApp.deiconify()
```

Funkcija koja se poziva na dva mesta. Prvo mesto koje poziva ovu funkciju je Exit dugme a drugo je `app.protocol("WM_DELETE_WINDOW", exitClick)`. app.protocol poziva ovu funkciju kada se isključi pogled klikom na X u gornjem desnom uglu prozora. Funkcija sakriva i gasi tabelu i vraća korisnika na početnu stranicu.

### undoClick()

```
def undoClick():
    global lastMove

    buttons[lastMove[0]][lastMove[1]].configure(fg_color = 'transparent',
state='normal')
    matrix[lastMove[0]][lastMove[1]] = eTilePlayable.value[0]
    undoButton.configure(state = 'disabled')
    confirmButton.configure(state = 'disabled')
    lastMove = None
```

Funkcija koja se poziva klikom na Undo dugme. Ova funkcija briše trenutno odabрано polje i onemogućava Undo i Confirm dugme iz istog razloga kao i klik na Confirm.

### getCurrentPlayerName()

```
def getCurrentPlayerName():
    if(players[currentPlayer] == eTilePlayer1.value[0]):
        return 'Player 1'
    elif(isBot):
        return 'Bot'
    else:
        return 'Player 2'
```

Ova funkcija vraća string vrednost trenutnog igrača korišćenjem enuma `eTile`.

### getCurrentPlayer()

```
def getCurrentPlayer():
    return f"Trenutni igrač: {getCurrentPlayerName()}"
```

Funkcija koja se koristi za label koji govori naziv trenutnog igrača. Ova funkcija poziva funkciju `getCurrentPlayerName()` kako se ne bi duplirao kod.

### render\_board(parent, matrix, rows, cols)

```
def render_board(parent, matrix, rows, cols):
    buttons = [[None for _ in range(cols)] for _ in range(rows)]

    for i in range(rows):
        for j in range(cols):
            if(matrix[i][j] == eTile.Invalid.value[0]):
                continue

            buttons[i][j] = create_button(parent, matrix[i][j], i, j)

    return buttons
```

Ova funkcija se koristi za štampanje table na kojoj se igra. Na samom početku kreira prazan niz koji će se kasnije popuniti dugmićima. Nakon toga prolazi kroz matricu koja je poslata kao parametar i ukoliko je poslato polje dobro kreira dugme za njega korišćenjem funkcije `create_button(parent, tile, row, col)`

Kao parametre ova funkcija ima:

*parent* – frame u kojem treba da se nalazi tabela, *matrix* – matrica koju generiše **createTable(n)** funkcija, *rows* – broj redova u dатој matrici, *cols* – broj kolona u dатој matrici

```
create_button(parent, tile, row, col)
```

```
def create_button(parent, tile, row, col):
    if tile == eTile.Playable.value[0]:
        fg = "transparent"
        state = "normal"
    elif tile == eTile.Player1.value[0]:
        fg = playerColors[0]
        state = "disabled"
    elif tile == eTile.Player2.value[0]:
        fg = playerColors[1]
        state = "disabled"

    btn = customtkinter.CTkButton(
        parent,
        text='',
        corner_radius=30,
        width=30,
        fg_color=fg,
        border_width=2,
        border_color=("##000", "#bab2b2"),
        state=state,
        command=lambda: on_tile_click(row, col)
    )

    btn.grid(row=row, column=col)
    return btn
```

Ova funkcija kreira dugme u polju matrica. Kao parametar prima Frame u kojem treba da se nalazi dugme i tip polja (enum `eTile`). Zavisno od poslatih vrednosti kreira odgovarajuće dugme sa odgovarajućom bojom i aktivnošću. Ukoliko je na datom polju neki igrač onda će dugme biti u boji tog igrača i biće disabled, u suprotnom će dugme biti providno i imati akciju na kliku (`on_tile_click(row, col)`).

### on\_tile\_click(r, c)

```
def on_tile_click(r, c):
    global lastMove, currentPlayer

    if matrix[r][c] != eTile.Playable.value[0]:
        return

    if lastMove != None:
        buttons[lastMove[0]][lastMove[1]].configure(fg_color = 'transparent',
state='normal')
        matrix[lastMove[0]][lastMove[1]] = eTile.Playable.value[0]

    lastMove = (r, c)
    undoButton.configure(state = 'normal')
    confirmButton.configure(state = 'normal')
    matrix[r][c] = players[currentPlayer]
    buttons[r][c].configure(fg_color = playerColors[currentPlayer],
state='disabled')
    print(f"{getCurrentPlayerName()} je odabrao: {chr(ord('A') - 1 + c)} {(r +
c - n + 1)//2}")
```

Funkcija koja se izvršava kada se klikne na određeno polje u listi slobodnih polja. Kao parametar ova funkcija ima red i kolonu u kojoj se dugme nalazi. Ukoliko ovo nije prvi potez trenutnog igrača, prethodno odigrano dugme se vraća u staro stanje dok novo dobija boju dok igrača. Na kraju funkcije se u terminalu štampa adresa dugmeta u matrici.

### changeAppTheme

```
def changeAppTheme():
    customtkinter.set_appearance_mode("dark" if
customtkinter.get_appearance_mode().lower() == "light" else "light")
```

Funkcija koja menja trenutnu temu aplikacije.

## enums.py

### eTile

```
class eTile(Enum):
    Invalid = -1,
    Playable = 0,
    Player1 = 1,
    Player2 = 2,
```

Enum kojim se obeležava tip polja u aplikaciji. Polja su Invalid za sve “prazne” vrednosti u matrici. To bi bila polja koja se ne prikazuju na samoj tabeli i koja su zapravo između dva polja.

## tableCreator.py

### createTable(n)

```
def createTable(n):
    matrixN = n * 4 - 1
    matrixM = n * 2 + 1
    matrix = createEmptyTable(matrixN, matrixM)
    matrix = fillTable(matrix, n, matrixN)
    return matrix, matrixN, matrixM
```

Ovo je jedina funkcija koja se poziva u drugom fajlu. Ona se poziva u fajlu **board.py** kako bi vratila matricu polja na kojima može i ne može da se igra. Kao parametar samo prihvata  $n$  što obeležava veličinu stranice.  $n$  mora biti neparan broj između 3 i 9. Funkcija određuje broj vrsti i broj kolana i dalje salje to ostalim funkcijama. Pre svega kreira praznu tabelu pozivom funkcije `createEmptyTable(matrixN, matrixM)` pa je nakon toga popunjuje pozivom funkcije `fillTable(matrix, n, matrixN)`. Funkcija kao povratne vrednosti ima popunjenu matricu i njene veličine.

```
createEmptyTable(matrixN, matrixM)
```

```
def createEmptyTable(matrixN, matrixM):
    res = []
    for _ in range(matrixN):
        row = []
        for _ in range(matrixM):
            row.append(eTile.Invalid.value[0])
        res.append(row)
    return res
```

Ova funkcija kao parametre prihvata veličinu matrice. Ona samo prolazi kroz matricu i popunjava je poljima koje imaju vrednost **eTile.Invalid**.

```
fillTable(matrix, n, matrixN)
```

```
def fillTable(matrix, n, matrixN):
    matrix = fillEdgeValues(matrix, n, matrixN)
    matrix = fillMiddleValues(matrix, n)
    return matrix
```

Ova funkcija kao parametar ima praznu matricu, veličinu stranice  $n$  i veličinu matrice. Ona kao rezultat ima popunjenu matricu sa vrednostima iz enuma **eTile**. Poziva druge dve funkcije koje popunjavaju odredjene delove matrice.

### fillEdgeValues(matrix, n, matrixN)

```
def fillEdgeValues(matrix, n, matrixN):
    for i in range(1, n):
        p1 = eTile.Player1.value[0] if i < n/2 else eTile.Player2.value[0]
        p2 = eTile.Player2.value[0] if i < n/2 else eTile.Player1.value[0]
        matrix[i - 1][n - i] = p1
        matrix[matrixN - i][n + i] = p1
        matrix[i - 1][n + i] = p2
        matrix[matrixN - i][n - i] = p2
        for j in range(n - i + 2, n + i - 1, 2):
            matrix[i - 1][j] = eTile.Playable.value[0]
            matrix[matrixN - i][j] = eTile.Playable.value[0]

    for i in range(2, n * 2 - 1, 2):
        matrix[n-1][i] = eTile.Playable.value[0]
        matrix[matrixN-n][i] = eTile.Playable.value[0]
    return matrix
```

Ova funkcija popunjava spoljašnje vrednosti matrice. Funkcija jednim prolazom popunjava i donju i gornju stranu matrice. Matrica je u suštini podeljena na 3 dela. Srednji deo bi bile simetrične vrednosti koje se nalaze na samoj sredini tabele što počinje sa poljima jednog igrača a završava se poljima drugog. Sa obzirom na to da su prvi i treći deo tabele isti, samo rotirani za 180 stepeni, moguće je popuniti obe strane prolazom kroz jednu.

### fillMiddleValues(matrix, n)

```
def fillMiddleValues(matrix, n):
    matrix = fillSeparator(matrix, n, n)
    crnt = 1
    for i in range(n+1, (n+1)*2+(n-4), 2):
        p1 = eTile.Player1.value[0] if crnt < n/2 else eTile.Player2.value[0]
        p2 = eTile.Player2.value[0] if crnt < n/2 else eTile.Player1.value[0]
        matrix[i][0] = p1
        matrix[i][n*2] = p2
        for j in range(2, n*2-1, 2):
            matrix[i][j] = eTile.Playable.value[0]
        matrix = fillSeparator(matrix, n, i+1)
        crnt+=1

    return matrix
```

Za razliku od *fillEdgeValues()* funkcije, ova funkcija popunjava drugi deo matrice, tačnije srednji deo. Jednim prolazom se popunjavaju dva reda u matrici, prvi red bi bio red koji počinje poljem igrača dok bi drugi red bio "prazan" red, tačnije red koji ne počinje poljem igrača. Za popunjavanje praznog polja se koristi funkcija *fillSeparator(matrix, n, i+1)*. Prolazi se kroz srednji deo i pomoću promenljive *crnt* se određuje da li je došlo do kraja prve polovine sredine, to se radi zbog toga što se na sredini menja početni igrač. Prvo se podese prvo i poslednje polje u matrici pa se nakon toga jednom for petljom prođe kroz sva parna polja gde se vrednosti postavljaju na **eTile.Playable**.

### fillSeparator(matrix, n, m)

```
def fillSeparator(matrix, n, m):
    for i in range(1, n*2, 2):
        matrix[m][i] = eTile.Playable.value[0]
    return matrix
```

Ova funkcija samo prolazi kroz sva neparna polja matrice i popunjava ih poljima koja su playable.

## INFORMACIJE O IGRI

### Tehnologije

Za kreiranje igre je korišćen programski jezik Python. Od spoljašnjih biblioteka je korišćena biblioteka **customtkinter** koja je iskorišćena za grafički interfejs igre. Ova biblioteka je nadogradnja postojeće python biblioteke **tkinter** koja sadrži osnovne funkcije za interfejs. Sve što **customtkinter** dodaje je lepši izgled takozvanim “widgetima” koji se koriste za prikaz elemenata. Sintaksa je dosta slična kreiranju web stranica korišćenjem DOM manipulacija u javascript-u. Druga biblioteka koja je korišćena je **PIL** koja je korišćena za dodavanje slika elementima. Konkretno je korišćena za dodavanje slika dugmićima na stranicama kao i kreiranje dugmeta za light i dark mod.

### Reference

**customtkinter** - <https://customtkinter.tomschimansky.com/documentation/>

**PIL** - <https://customtkinter.tomschimansky.com/documentation/utility-classes/image>