

Poglavlje 15

Alat za generisanje dokumentacije

15.1 Uvod

Dokumentacija često predstavlja stavku koja se zanemaruje prilikom rada na nekom projektu. U nekim specijalnim situacijama, ukoliko postoje određeni ograničavajući faktori (na primer vreme), ovo se može smatrati i opravdanim. Ipak, u opštem slučaju, dokumentovanje projekta treba da bude stvar koja se *de facto* podrazumeva. Razlog za to je sledeći.

Naime, dokumentacija prati razvoj svakog projekta i, ujedno, povećava njegov kvalitet. Dobro napisana dokumentacija pomaže prilikom, kako razvoja, tako i održavanja nekog projekta ili već gotovog proizvoda. Prema nekim procenama, održavanje nekog softverskog proizvoda oduzima i do *tri* puta više vremena od samog razvoja datog proizvoda, u okviru njegovog životnog veka. U tom procesu održavanja, čitanje već napisanog koda predstavlja komponentu koja programeru oduzima najviše vremena.

Pored toga, još jedna važna uloga dokumentacije jeste i u *prenosu znanja*. Naime, broj različitih ljudi koji trenutno rade na nekom projektu je varijabilan. Za nove programere je potrebno da, u što je moguće kraćem vremenskom roku, dobiju potpunu sliku o samom projektu. Dobra dokumentacija značajno ubrzava ovu proceduru.

Prilikom rada na softverskim projektima, pod dokumentacijom se najčešće podrazumeva dokumentacija koda. Prilikom dokumentovanja delova koda, generalna preporuka jeste dati odgovor na naredna tri pitanja.

- Koja je uloga datog dela koda?
- Kako dati deo koda izvršava tu ulogu?
- Kako neko može iskoristiti dati deo koda na nekom drugom mestu?

Kako bi se sama procedura dokumentovanja olakšala i ubrzala, u praksi postoje različiti alati za automatsko generisanje dokumentacije. Jedan od takvih alata jeste i *Doxygen*, čije objašnjenje sledi u nastavku.

15.2 Kreiranje dokumentacije korišćenjem alata Doxygen

Doxygen predstavlja alat otvorenog koda koji omogućava automatizovano generisanje projektne dokumentacije. Dokumentacija, pri tome, mora biti sadržana unutar samog izvornog koda, u obliku komentara koji poštuju format koji će biti opisan kasnije u tekstu.

Alat je moguće besplatno preuzeti sa zvanične *Doxygen* stranice, putem sledećeg linka:

<https://www.doxygen.nl/index.html>

Sam alat je moguće koristiti na dva različita načina:

- iz komandne linije
- pomoću *Doxywizard*-a

Rad sa *Doxygen* alatom iz komandne linije

Standardna procedura generisanja dokumentacije upotrebom alata iz komandne linije je sledeća:

1. Potrebno je dokumentovati izvorni kod u skladu sa odgovarajućim pravilima.
2. Potrebno je generisati konfiguracionu datoteku pozivom alata sa odgovarajućom opcijom, na sledeći način:

```
<putanja do .exe datoteke>\doxygen.exe -g <konfiguraciona datoteka>
```

gde parametar `<konfiguraciona datoteka>` predstavlja naziv datoteke koja će biti generisana.

3. Potrebno je modifikovati parametre unutar konfiguracione datoteke u skladu sa projektom za koji se generiše dokumentacija. Detaljno objašnjenje svih parametara je moguće pronaći na narednom linku:

<https://www.doxygen.nl/manual/config.html>

4. Konačno, potrebno je generisati dokumentaciju korišćenjem alata, na sledeći način:

```
<putanja do .exe datoteke>\doxygen.exe <konfiguraciona  
datoteka>
```

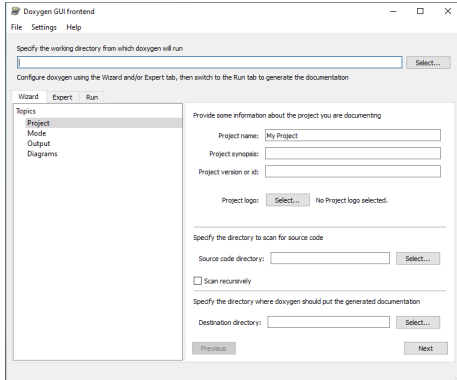
gde parametar <konfiguraciona datoteka> predstavlja naziv prethodno modifikovane konfiguracione datoteke.

Korisna opcija jeste i opcija `--help`, koja omogućava kratko objašnjenje načina na koje se alat može koristiti iz komandne linije.

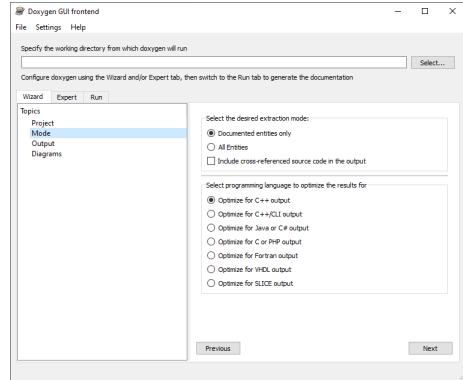
Rad sa *Doxygen* alatom korišćenjem *Doxywizard-a*

Drugi način rada sa *Doxygen* alatom podrazumeva upotrebu ugrađenog *Doxywizard-a*. Ovaj pristup je značajno lakši za korišćenje od prethodnog, te se, u praksi, uglavnom i češće koristi. Navigacijom kroz prozore *wizard-a*, na jednostavan način je moguće popuniti sve neophodne parametre koji se tiču trenutnog projekta, kao što su naziv projekta, ulazni direktorijum u okviru kog je smešten dokumentovani izvorni kod, izlazni format i drugo.

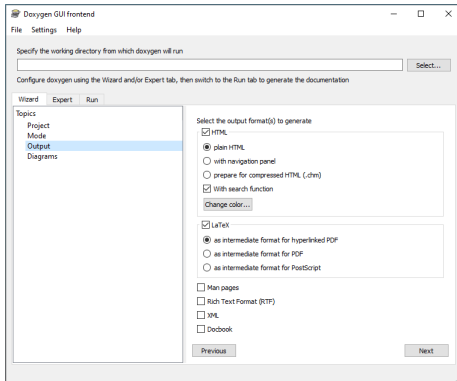
Na narednim slikama, prikazani su osnovni delovi *Doxywizard-a*.



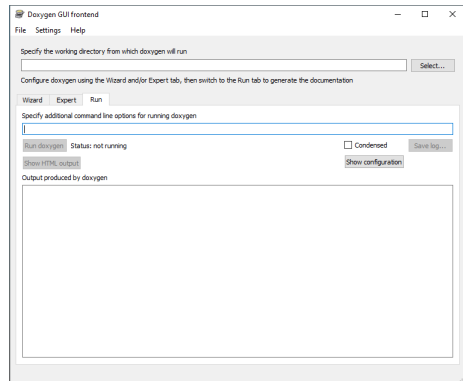
(a) Karakteristike projekta



(b) Izbor režima izvršavanja



(c) Izbor izlaznog formata



(d) Run meni

Slika 15.1: Izgled Doxywizard-a

Detaljno objašnjenje *Doxywizard*-a je moguće pronaći na narednom linku:

https://www.doxygen.nl/manual/doxywizard_usage.html

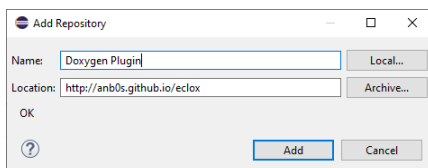
Rad sa *Doxygen* alatom u okviru razvojnog okruženja *Eclipse*

Pored navedena dva, standardna načina, razvojno okruženje *Eclipse*, takođe podržava rad sa ovim alatom preko odgovarajućeg dodatka (eng. *plugin*) zvanog *eclox*.

Instalacija *eclox* dodatka vrši se izborom opcije *Help* → *Install New Software* u okviru *Eclipse* okruženja. Po otvaranju prozora, u polju *Work with* potrebno je upisati sledeći link:

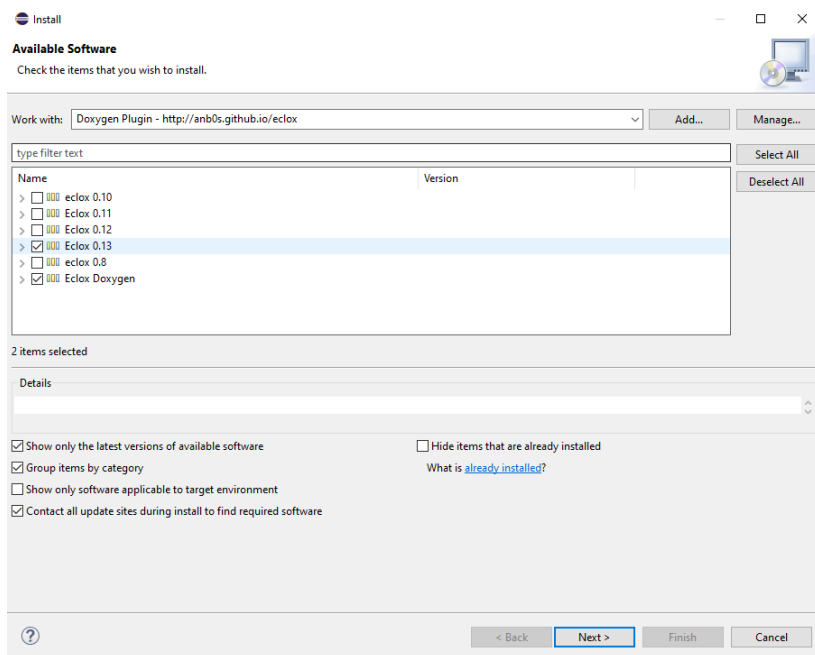
<http://anb0s.github.io/eclox>

i kliknuti *Add*. Po iskakanju prozora *Add Repository*, u polju *Name* dodeljuje se ime *eclox* dodatku, na primer, *Doxygen Plugin*. Ovo je prikazano na slici 15.2.



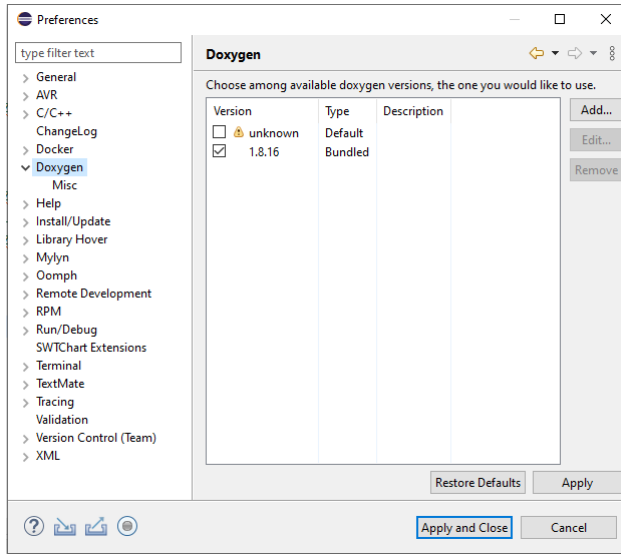
Slika 15.2: Prozor *Add Repository*

Nakon toga, potrebno je selektovati najnoviju dostupnu verziju *eclox* dodatka i *Eclox Doxygen*. Ovo je ilustrovano na slici 15.3. Nakon toga, potrebno je ispratiti preostale korake do završetka instalacije.



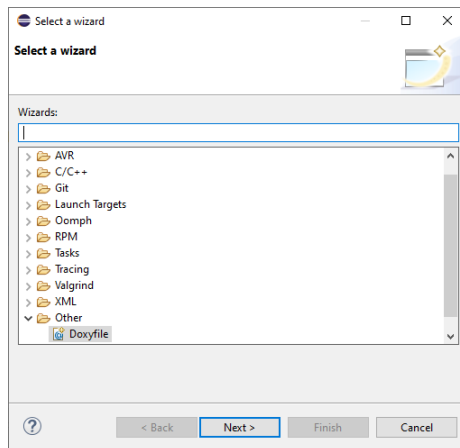
Slika 15.3: Izbor dodatka u okviru menija za izbor dodataka

Nakon instalacije *eclox* dodatka, potrebno je podesiti putanju do izvršne verzije *Doxygen*-a, pomoću opcije *Window* → *Preferences* → *Doxygen*, kao na slici 15.4.



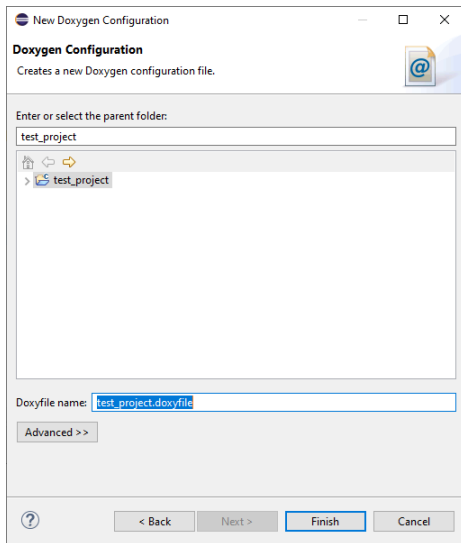
Slika 15.4: Podešavanje putanje do izvršne verzije Doxygen-a

Kako bi bilo omogućeno korišćenje *Doxygen*-a, u okviru projekta potrebno je dodati konfiguracionu datoteku, pomoću opcije *New* → *Other* → *Other* → *Doxyfile*. Ovo je ilustrirano na slici 15.5.



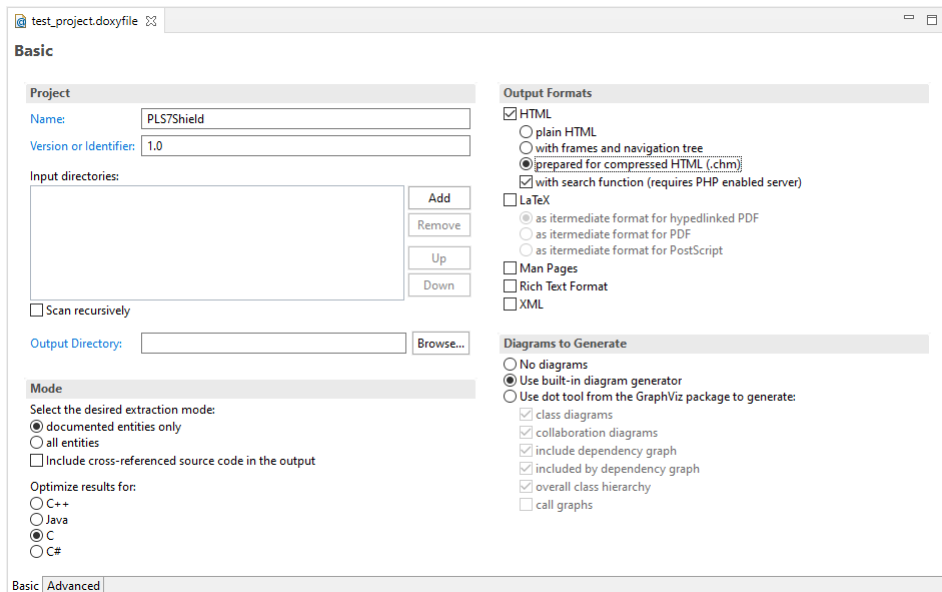
Slika 15.5: Dodavanje konfiguracione datoteke

Nakon toga je potrebno podesiti putanju do direktorijuma u kojem će biti kreirane *Doxygen* datoteke. Takođe, potrebno je uneti ime *.doxyfile* datoteke, kao na slici 15.6.



Slika 15.6: Podešavanje putanje i unos imena `.doxyfile` datoteke

Dvoklikom na ime `.doxyfile` datoteke, otvara se prozor u kojem se definišu opcije kao što su naziv projekta, ulazni direktorijum (koji sadrži datoteke koje će biti skenirane), izlazni direktorijum koji će sadržati generisanu dokumentaciju, izlazni format i drugo. Konfiguracioni prozor Doxygen datoteke je prikazan na slici 15.7.



Slika 15.7: Konfiguracioni prozor `.doxyfile` datoteke

Dokumentacija se generiše pomoću opcije *Build Documentation* u izabranom formatu i nalazi se u odgovarajućem izlaznom folderu. U nastavku je objašnjen način pisanja komentara prilagođenih ovom alatu.

15.3 Stil pisanja komentara za Doxygen

Prilikom generisanja dokumentacije, *Doxygen* skenira izvorne .h i .c datoteke u ulaznom direktorijumu i pri tome traži komentare koji su propisno formatirani. Na primer, zaglavlje datoteke može imati format prikazan na narednom listingu.

```
/**
 * @file ime_datoteke.c
 * @brief Opis datoteke
 * @author Ime Prezime
 * @date DD-MM-YYYY
 * @version 1.0
 */
```

Listing 15.1: Validan format dokumentovanja zaglavlja

U okviru zaglavlja, svaka linija komentara počinje karakterom '*'. Polja se označavaju ključnim rečima kojima prethodi karakter '@' (ili '\'), nakon čega se navodi njihov opis. Listu svih dostupnih ključnih reči je moguće pronaći na sledećem linku.

<https://www.doxygen.nl/manual/commands.html>

Njihovo objašnjenje daleko prevazilazi okvire ovog praktikuma, te će na ovom mestu biti izostavljeno. Svakako, zainteresovanom čitaocu se preporučuje da ih detaljnije prouči.

Ukoliko korisnik unese "/**" u redu koji prethodi prototipu funkcije i nakon toga pritisne taster *Enter*, Doxygen će automatski generisati blok za opis funkcije:

```
/**
 * @param str
 * @param n
 * @return
 */
int16_t myFunction(int8_t* str, int16_t n)
```

Listing 15.2: Validan format dokumentovanja funkcije

Polja @param sadrže opise ulaznih argumenata, dok polje @return opis povratne vrednosti.

Doxygen uzima u obzir i jednolinijske komentare koji počinju sa tri kose crte – "///", kao što je i prikazano na listingu 15.3.


```
/// Primer jednolinijskog komentara
int16_t var;
```

Listing 15.3: Validan format linijskog komentara

U nastavku je, u okviru Primera 15.3.1, dat izvorni kod prilagođen generisanju dokumentacije pomoću *Doxygen*-a.

Primer 15.3.1. Napisati program koji implementira treperenje ugrađene diode na Arduino platformi. Izvorni kod programa je potrebno dokumentovati u skladu sa prethodno opisanim pravilima.

Rešenje:

```
/**
 * @file main.c
 * @brief Aplikacija koja implementira ograničen broj treptanja
 *        diode
 * @author Ime Prezime
 * @date 14-03-2021
 * @version 1.0
 */
#include <avr/io.h>
#include <avr/interrupt.h>
#include <stdint.h>

/// Makro za podesavanje visoke vrednosti signala na pinu
#define HIGH 1
/// Makro za podesavanje niske vrednosti signala na pinu
#define LOW 0

/// Makro za podesavanje izlaznog smera pina
#define OUTPUT 1
/// Makro za podesavanje ulaznog smera pina
#define INPUT 0

/// Makro za selektovanje porta B
#define PORT_B 0
/// Makro za selektovanje porta C
#define PORT_C 1
/// Makro za selektovanje porta D
#define PORT_D 2

/// Makro za selektovanje pina na koji je povezana dioda
#define DIODE_PIN 5

/// Promenljiva koja skladišti broj milisekundi proteklih od
/// pokretanja aplikacije
volatile uint32_t ms = 0;
```

```

/**
 * pinPulse - Funkcija koja implementiran podizanje i spustanje
 *            vrednosti na pinu
 *            odgovarajucom brzinom
 * @param port - ulaz tipa uint8_t - Port na kojem je potrebno
 *            implementirati funkcionalnost
 * @param pin - ulaz tipa uint8_t - Pin na kojem je potrebno
 *            implementirati funkcionalnost
 * @param period - ulaz tipa uint32_t - Perioda promene
 *            vrednosti na pinu
 * @return Nema povratnu vrednost
 */
void pinPulse(uint8_t port, uint8_t pin, uint32_t period);

/**
 * pinSetValue - Funkcija koja postavlja vrednost na pinu
 * @param port - ulaz tipa uint8_t - Port na kojem je pin ciju
 *            vrednost potrebno postaviti
 * @param pin - ulaz tipa uint8_t - Pin ciju je vrednost
 *            potrebno postaviti
 * @param value - ulaz tipa uint8_t - Vrednost koju je potrebno
 *            postaviti na pin
 * @return Nema povratnu vrednost
 */
void pinSetValue(uint8_t port, uint8_t pin, uint8_t value);

/**
 * pinInit - Funkcija koja implementiran inicijalizaciju pina
 * @param port - ulaz tipa uint8_t - Port na kojem je pin koji
 *            je potrebno inicijalizovati
 * @param pin - ulaz tipa uint8_t - Pin koji je potrebno
 *            inicijalizovati
 * @param direction - ulaz tipa uint8_t - Smer prema kojem je
 *            potrebno inicijalizovati pin
 * @return Nema povratnu vrednost
 */
void pinInit(uint8_t port, uint8_t pin, uint8_t direction);

/**
 * timer0DelayMs - Funkcija koja implementira pauzu u broju
 *            milisekundi koji je prosledjen
 *            kao parametar
 * @param delay_length - ulaz tipa uint32_t - Duzina pauze u
 *            milisekundama
 * @return Nema povratnu vrednost
 */
void timer0DelayMs(uint32_t delay_length);

/**
 * timer0Millis - Funkcija koja, na bezbedan nacin, vraca kao

```

```
    povratnu vrednost broj milisekundi proteklih od pocetka
    merenja vremena
* @return Broj milisekundi proteklih od pocetka merenja
    vremena
*/
uint32_t timer0Millis();

/**
* timer0Init - Funkcija koja inicijalizuje timer 0 tako da
    pravi prekide
* svake milisekunde
* @return Nema povratnu vrednost
*/
void timer0Init();

/**
* main - funkcija koja implementiran glavni deo aplikacije
* @return Nema povratnu vrednost
*/
int16_t main()
{
    uint32_t period = 1000;        // Period jednog treptaja
    uint8_t repetitions = 5;      // Broj treptaja
    int16_t i;

    // Inicijalizacija
    pinInit(PORT_B, DIODE_PIN, OUTPUT);
    timer0Init();

    // Glavna petlja
    while (1)
    {
        // Treptanje
        for (i = 0; i < repetitions; i++)
            pinPulse(PORT_B, DIODE_PIN, period);

        // Kraj
        while (1)
            ;

    }

    return 0;
}

/*****

void pinPulse(uint8_t port, uint8_t pin, uint32_t period)
{
    // Poluperioda u kojoj pin ima visoku vrednost
```

```

pinSetValue(port, pin, HIGH);
timer0DelayMs(period/2);

// Poluperioda u kojoj pin ima nisku vrednost
pinSetValue(port, pin, LOW);
timer0DelayMs(period/2);
}

/*****/

void pinSetValue(uint8_t port, uint8_t pin, uint8_t value)
{
    // Postavljanje vrednosti pina
    switch(port)
    {
        case PORT_B:
            if (value == HIGH)
                PORTB |= 1 << pin;
            else
                PORTB &= ~(1 << pin);
            break;

        case PORT_C:
            if (value == HIGH)
                PORTC |= 1 << pin;
            else
                PORTC &= ~(1 << pin);
            break;

        case PORT_D:
            if (value == HIGH)
                PORTD |= 1 << pin;
            else
                PORTD &= ~(1 << pin);
            break;

        default:
            break;
    }
}

/*****/

void pinInit(uint8_t port, uint8_t pin, uint8_t direction)
{
    // Inicijalizacija smeru pina
    switch (port)
    {
        case PORT_B:
            if (direction == OUTPUT)

```

```
        DDRB |= 1 << pin;
    else
        DDRB &= ~(1 << pin);
    break;

    case PORT_C:
        if (direction == OUTPUT)
            DDRC |= 1 << pin;
        else
            DDRC &= ~(1 << pin);
        break;

    case PORT_D:
        if (direction == OUTPUT)
            DDRD |= 1 << pin;
        else
            DDRD &= ~(1 << pin);
        break;

    default:
        break;
}

}

/*****/

void timer0DelayMs(uint32_t delay_length)
{
    // trenutak pocevsi od kog se racuna pauza
    uint32_t t0 = timer0Millis();
    // implementacija pauze
    while(timer0Millis() - t0 < delay_length)
        ;
}

/*****/

uint32_t timer0Millis()
{
    uint32_t tmp;
    cli();           // Zabrana prekida
    tmp = ms;       // Ocitanje vremena
    sei();          // Dozvola prekida
    return tmp;
}

/*****/

void timer0Init()
```

```

{
    TCCR0A = 0x02; // tajmer 0: CTC mod

    // Provera frekvencije takta prilikom kompilacije
    #if F_CPU > 20000000
    #error "Frekvencija takta mora biti manja od 20MHz!"
    #endif

    // Inicijalizacija promenljivih za preskaler i periodu
    // tajmera 0
    uint32_t n = F_CPU / 1000;
    uint8_t clkssel = 1;

    // Odredjivanje vrednosti preskalera i periode tajmera 0
    while (n > 255)
    {
        n /= 8;
        clkssel++;
    }

    TCCR0B = clkssel; // tajmer 0: Podesavanje preskalera
    OCR0A = (uint8_t)(n & 0xff) - 1; // tajmer 0: Podesavanje
    // periode
    TIMSK0 = 0x02; // tajmer 0: Dozvola prekida
    sei(); // Globalna dozvola prekida
}

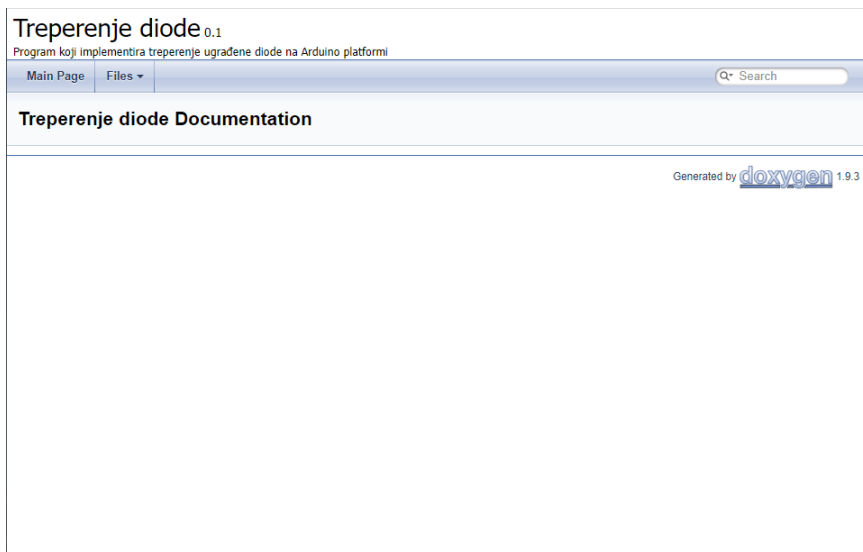
/*****

/**
 * ISR - prekidna rutina tajmera 0 u modu CTC
 */
ISR(TIMERO_COMPA_vect)
{
    // Inkrementovanje broja milisekundi koje su prole od
    // pokretanja aplikacije
    ms++;
}

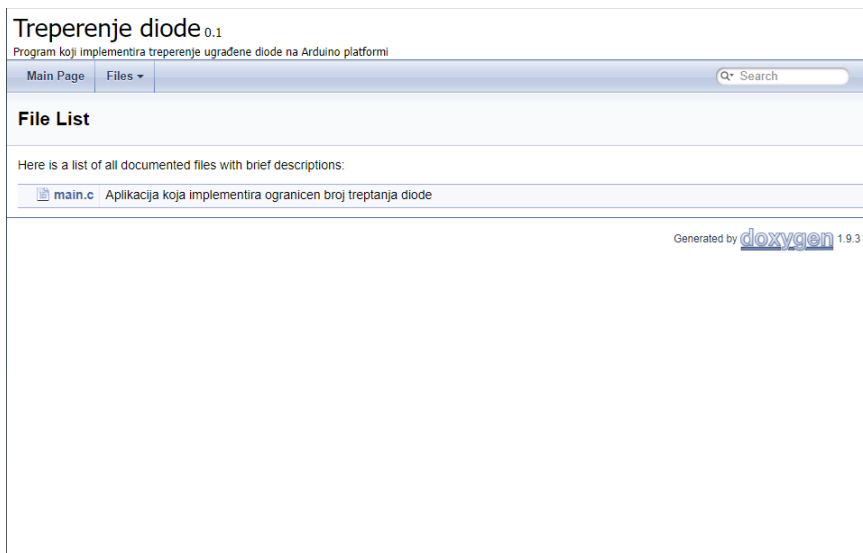
```

Listing 15.4: Izvorni kod sa Doxygen komentarima

Generisanoj dokumentaciji, u HTML formatu, je moguće pristupi otvaranjem *index.html* stranice, u proizvoljnom internet pretraživaču. Izgled ove stranice za prethodni primer je dat na slikama 15.8, 15.9 i 15.10.



Slika 15.8: Stranica *Index.html*



Slika 15.9: Lista izvornih datoteka

Treperenje diode 0.1
Program koji implementira treperenje ugrađene diode na Arduino platformi

Main Page | Files ▾ | Search

Macros | Functions | Variables

main.c File Reference

Aplikacija koja implementira ograničen broj treptanja diode. More...

```
#include <avr/io.h>
#include <avr/interrupt.h>
#include <stdint.h>
```

Macros

#define HIGH 1	Makro za podesavanje visoke vrednosti signala na pinu.
#define LOW 0	Makro za podesavanje niske vrednosti signala na pinu.
#define OUTPUT 1	Makro za podesavanje izlaznog smera pina.
#define INPUT 0	Makro za podesavanje ulaznog smera pina.
#define PORT_B 0	Makro za selektovanje porta B.
#define PORT_C 1	Makro za selektovanje porta C.
#define PORT_D 2	Makro za selektovanje porta D.
#define DIODE_PIN 5	Makro za selektovanje pina na koji je povezana dioda.

Functions

void	pinPulse (uint8_t port, uint8_t pin, uint32_t period)
void	pinSetValue (uint8_t port, uint8_t pin, uint8_t value)
void	pinInit (uint8_t port, uint8_t pin, uint8_t direction)
void	timer0DelayMs (uint32_t delay_length)
uint32_t	timer0Millis ()
void	timer0Init ()
int16_t	main ()

Slika 15.10: Dokumentacija datoteke main.c 1

15.4 Zadatak za vežbu

Zadatak 15.4.1. Dokumentovati, u skladu sa *Doxygen* pravilima, **pet** projekata (uključujući i korišćene korisničke biblioteke), realizovanih u prethodnim poglavljima, po izboru. Upotrebom alata *Doxygen* izvršiti generisanje dokumentacije, na naredna tri načina:

- upotrebom alata iz komandne linije;
- upotrebom *Doxywizard*-a;
- upotrebom *eclox* dodatka u okviru razvojnog okruženja *Eclipse*.