



Thingspeak IoT platforma

ThingSpeak

- IoT analitička platforma koja omogućuje prikupljanje, vizuelizaciju i analizu padataka uživo.
- ThingSpeak omogućuje inženjerima i naučnicima da naprave prototip i IoT sistem bez postavljanja servera i razvijanja web softvera.
- Otvorena IoT platforma sa MATLAB analitikom
- Nalazi se na adresi: [IoT Analytics - ThingSpeak Internet of Things](#)

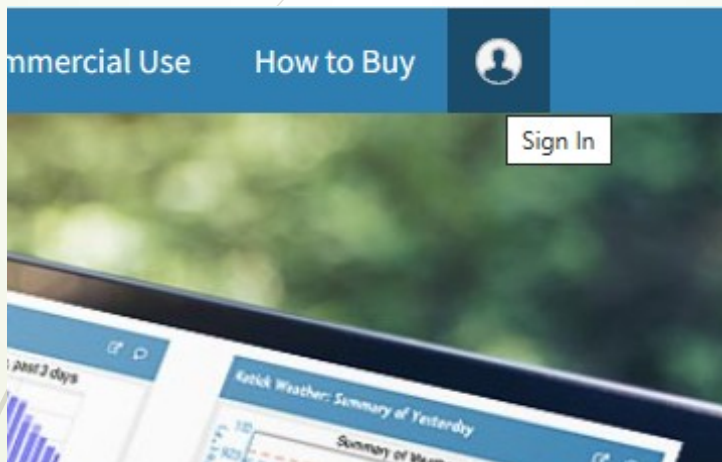
ThingSpeak™ Channels Apps Support Commercial Use How to Buy

ThingSpeak for IoT Projects

Data collection in the cloud with advanced data analysis using MATLAB

[Get Started For Free](#) [Learn More](#)

ThingSpeak prijavljivanje

The MathWorks login page. It features the MathWorks logo at the top. Below it is an 'Email' label and an empty input field. Under the input field, there's a link that says 'No account? Create one!'. Below this link is a line of text: 'By signing in, you agree to our [privacy policy](#).' At the bottom right of the form is a blue 'Next' button.The 'Create MathWorks Account' form. It starts with the title 'Create MathWorks Account'. The first section is 'Email Address', which has an empty input field with a red exclamation mark icon to its right. Below the input field is a red error message box that says 'Missing required information'. Below this is an information icon followed by the text: 'To access your organization's MATLAB license, use your school or work email.' The next section is 'Location', which has a dropdown menu currently showing 'Montenegro'. Below that are two more input fields labeled 'First Name' and 'Last Name'. At the bottom of the form are two buttons: a blue 'Continue' button and a white 'Cancel' button with a blue border.

Kreiraj račun na ThingSpeak <https://thingspeak.com/>

HTTP – za komuniciranje sa ThingSpeak

- HTTP – Hypertext Transfer protokol
 - Dizajniran da omogući komunikaciju između servera i klijenta
 - Protokol zahtjeva i odgovora
 - Klijent šalje HTTP zahjev serveru – server klijentu uzvraća odgovor
 - Odgovor sadrži status izvršenja zahtjeva, a može sadržati i dodatne podatke.
-
- U radu sa ThingSpeak platformom Arduino uređaj će imati ulogu klijenta a ThingSpeak platforma ulogu servera.



HTTP zahtjev

HTTP zahtjev generiše klijent, prema imanovanom host-u, lociranom na serveru.

Cilj zahtjeva je pristup resursu na serveru.

Korektno sastavljen HTTP zahtjev sadrži sljedeće elemente:

- Liniju zahtjeva;
- HTTP zaglavlja;
- Tijelo poruke, ako je potrebno.

Nakon svakog HTTP zaglavlja slijedi znak za povratak na početak reda (carriage return) i znak za prelazak u novi red (line feed) (CR-LF). Nakon posljednjeg zaglavlja dodatni CR-LF je dodat (za dobijanje prazne linije), nakon kojeg počinje tijelo poruke.

HTTP zahtjev – Linija zahtjeva

Linija zaglavlja je prva linija u poruci zahtjeva. Sastoji se iz tri dijela:

- ▢ Metod. Metod je jedno-rječna komanda koja govori serveru što da radi sa resursom. Na primjer, server može biti upitan da pošalje resurs klijentu.
- ▢ Komponenta staze URL-a za zahtjev. Staza identifikuje resurs na serveru.
- ▢ Broj HTTP verzije, ukazuje na HTTP specifikaciju s kojom je klijent pokušao uskladiti poruku.

Primjer linije zahtjeva:

GET /software/http/cic/indeks.html HTTP/1.1

Linija zahtjeva može sadržati i dodatne podatke.

HTTP zahtjev – Zaglavlje (Header)

- ▢ Pruža prijemnoj strani informacije o poruci, pošiljaocu i načinu na koji pošiljaoc želi da komunicira sa primaocem.
- ▢ Svako HTTP zaglavlje se sastoji od imena i vrijednosti.
- ▢ HTTP protokol definiše standardni set HTTP zaglavlja i opisuje kako ih koristiti korektno.
- ▢ HTTP zaglavlje zahtjeva klijenta sadrži informacije koje server može upotrijebiti u odlučivanju kako da odgovori na zahtjev. To može biti da klijent čita zahtijevani dokument na francuskom ili njemačkom jeziku i da dokument treba biti poslat jedino ako je mijenjan od naznačenog datuma.

```
Accept-Language: fr, de  
If-Modified-Since: Fri, 10 Dec 2004 11:22:13 GMT
```



HTTP zahtjev – Tijelo poruke

- ▢ Može se nazvati i tijelom zahtjeva
- ▢ Aktuelni sadržaj poruke.
- ▢ Tijelo poruke može biti u originalnom obliku ili može biti kodirano.
- ▢ Može se nazvati i tijelom zahtjeva
- ▢ Prikladno je za neke metode zahtjeva, dok za druge nije.
- ▢ Na primjer, zahtjev sa POST metodom, koji šalje ulazne podatke serveru, ima tijelo poruke, koje sadrži te podatke.
- ▢ Zahtjev sa GET metodom, koji od servera traži da pošalje resurs, ne sadrži tijelo poruke.



HTTP odgovor

- HTTP odgovor generiše server i šalje klijentu.
- Cilj odgovora je da obezbijedi klijentu treženi resurs ili da ga informiše o izvršenju zahtjeva ili da dojavu da je došlo do greške.
- HTTP odgovor se sastoji iz:
 - Statusne linije;
 - Zaglavlja;
 - Tijela poruke, koje je obično neophodno.

Nakon svakog HTTP zaglavlja slijedi znak za povratak na početak reda (carriage return) i znak za prelazak u novi red (line feed) (CR-LF). Nakon posljednjeg zaglavlja dodatni CR-LF je dodat (za dobijanje prazne linije), nakon kojeg počinje tijelo poruke.

HTTP odgovor - Statusna linija

- Statusna linija je prva linija u odgovoru. Sasloji se iz tri segmenta:
 - Broj HTTP verzije, koji ukazuje na HTTP specifikaciju po kojoj je server pokušao da uskladi odgovor.
 - Statusni kod, koji je trocifarski broj i ukazuje na rezultat izvršenja zahtjeva.
 - Fraza razloga, poznata i kao tekst statusa, koji je čitljiv čovjeku i sažima značenje statusnog koda.

Primjer statusne linije:

```
HTTP/1.1 200 OK
```

HTTP odgovor – Zaglavlja (Headers)

- ▮ Sadrži informacije koje klijent koristi da pronađe više podataka o odgovoru, kao i da pronađe podatke o serveru koji je poslao poruku.
- ▮ Ove informacije mogu pomoći klijentu u prezentaciji odgovora korisniku.
- ▮ Na primjer, prikazana zaglavlja govore klijentu kada je odgovor poslat, od strane kojeg servera je poslat, kao i da je to JPEG slika.

```
Date: Thu, 09 Dec 2004 12:07:48 GMT  
Server: IBM_CICS_Transaction_Server/3.1.0(zOS)  
Content-type: image/jpg
```

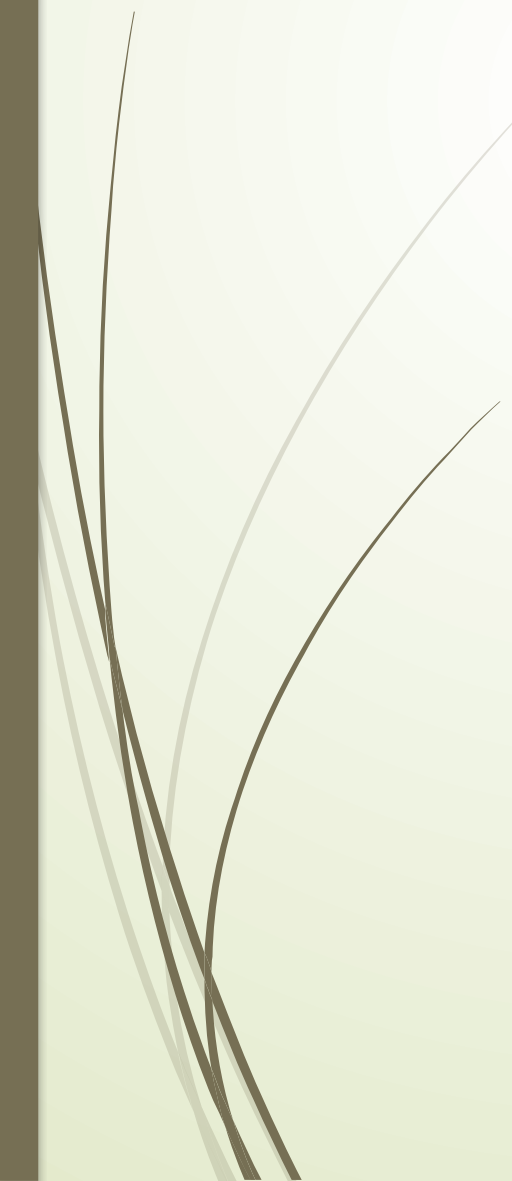


HTTP odgovor – Tijelo poruke

- ▮ Naziva se i tijelom odgovora.
- ▮ Većina odgovora sadrže tijelo poruke. Izuzeci su kada server odgovara na zahtjev klijenta, koji je koristio HEAD metod (koji koristi zaglavlja ali ne i tijelo odgovora) i gdje server koristi određene statusne kodove.
- ▮ U odgovoru na uspješno izvršen zahtjev, tijelo poruke sadrži resurs koji je klijent zahtijevao ili neke informacije o statusu radnje koji je klijent zahtijevao.
- ▮ U odgovoru na neuspješno izvršen zahtjev, tijelo poruke može da pruži dodatne informacije o razlozima greške ili o nekoj radnji koju klijent treba da preduzima da bi se zahtjev uspješno izvršio.



HTTP zahtjev - metode

- HTTP definiše set metoda (načina) da indicira akciju koja će biti izvršena na datom resursu.
 - Mada mogu biti i imenice, metode zahtjeva se često označavaju kao HTTP glagoli.
 - Svaki metod koristi različitu semantiku.
- 

HTTP zahtjev – vrste metoda

- ❑ **GET** Get metod se koristi za zahtijevanje podataka iz preciziranog resursa.
- ❑ **HEAD** HEAD metod zahtijeva zaglavlja resursa identificiranog datim URL-om, bez vraćanja samog resursa.
- ❑ **POST** POST metod se koristi za slanje podataka na obradu određenom resursu, često izazivajući promjene stanja ili druge efekte.
- ❑ **PUT** PUT se koristi za ažuriranje ili zamjenu postojećeg resursa na serveru sa obezbijeđenim podacima.
- ❑ **DELETE** metod briše specificirani resurs.
- ❑ **CONNECT** Uspostavlja vezu sa serverom, onosno, identificiranim ciljanim resursom.
- ❑ **OPTIONS** metod u HTTP protokolu koristi se za traženje informacija o komunikacijskim opcijama dostupnim za određeni resurs ili server. Primarna uloga OPTIONS metoda je osigurati metapodatke o mogućnostima i konfiguraciji servera ili resursa.
- ❑ **TRACE** metod u HTTP protokolu koristi se u dijagnostičke svrhe i obično se ne koristi u uobičajenim scenarijima web aplikacija. Njegova primarna uloga je osigurati način na koji klijent može zatražiti ako ih je da server vrati primljenu poruku zahtjeva, dopuštajući klijentu da vidi koje su promjene, bilo, usput napravili posrednički serveri.
- ❑ **PATCH** metod obavlja parcijalnu modifikaciju resursa

HTTP zahtjev – **POST** metod

- ❑ POST metod se koristi za slanje podataka serveru za kreiranje/ažuriranje resursa
- ❑ Podaci poslani serveru POST metodom smješteni su u tijelu HTTP zahtjeva.

POST /test/demo_form.php HTTP/1.1
Host: w3schools.com

name1=value1&name2=value2

- ❑ Nekoliko napomena u vezi POST zahtjeva:
 - ❑ POST zahtevi se nikada ne kešuju
 - ❑ POST zahtevi ne ostaju u historiji pregledača
 - ❑ POST zahtevi nemaju ograničenja u pogledu dužine podataka

HTTP zahtjev – **GET** metod

- GET se koristi za traženje podataka iz specificiranog izvora
- Treba imati na umu da se upitni string (par ime/vrijednost) šalje u URL-u GET zahtjeva.

/test/demo_form.php?name1=value1&name2=value2

- Nekoliko napomena u vezi GET zahtjeva:
 - GET zahtjevi se mogu keširati (spremiti u predmemoriju)
 - GET zahtjevi ostaju u istoriji pregledača
 - GET zahtjevi se nikada ne bi trebali koristiti kada se radi o osjetljivim podacima
 - GET zahtjevi imaju ograničenje dužine
 - GET zahtjevi se koriste samo za traženje podataka (ne promjenu)

HTTP zahtjev – GET vs. POST metod

	GET	POST
Keširanje	Može se keširati	Ne može se keširati
Tip kodiranja	application/x-www-form-urlencoded	application/x-www-form-urlencoded ili multipart/form-data. Upotreba višedjelnog kodiranja za binarne podatke
Istorija	Parametri ostaju u istoriji pregledača	Parametri ne ostaju u istoriji pregledača
Ograničenja u dužini podataka	Da, kod slanja podataka, GET method dodaje podatke na URL; dužina URL-a je ograničena (maximalna URL dužina je 2048 karaktera)	Bez ograničenja
Ograničenja u tipu podataka	Samo ASCII karakteri dozvoljeni	Bez restrikcija. Binarni podaci su takođe dozvoljeni
Bezbjednost	GET manje siguran u poređenju s POST, jer su podaci dio URL-a Ne koristiti GET kada se šalje lozinka ili druge osjetljive informacije	POST je malo sigurniji od GET jer parametri nijesu smješteni u istoriji pregledača ili u web server logu
Vidljivost	Podaci su vidljivi svima u URL-u	Podaci nijesu prikazani u URL-u

Arduino UNO+ESP8266 – GET preko URL-a

URL (Uniform Resource Identifier) je u stvari web adresa oblika http ili https.



HTTP GET

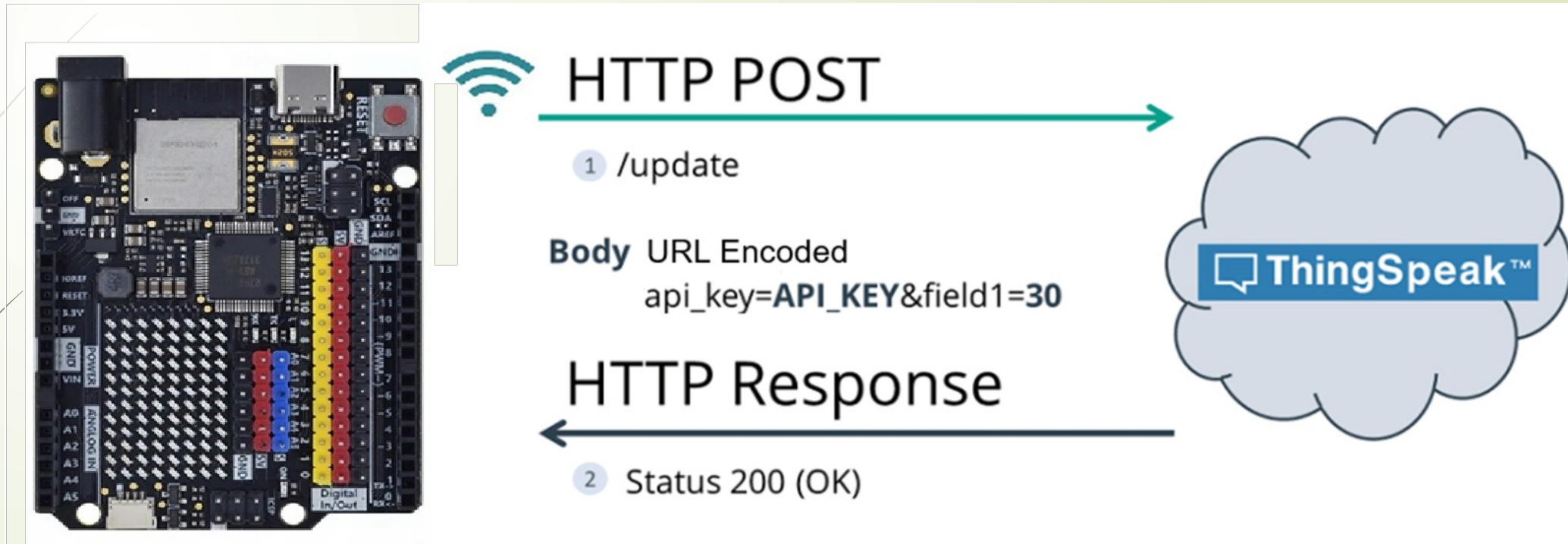
1 /update?api_key=**API_KEY**&field1=30

HTTP Response

2 Status 200 (OK)

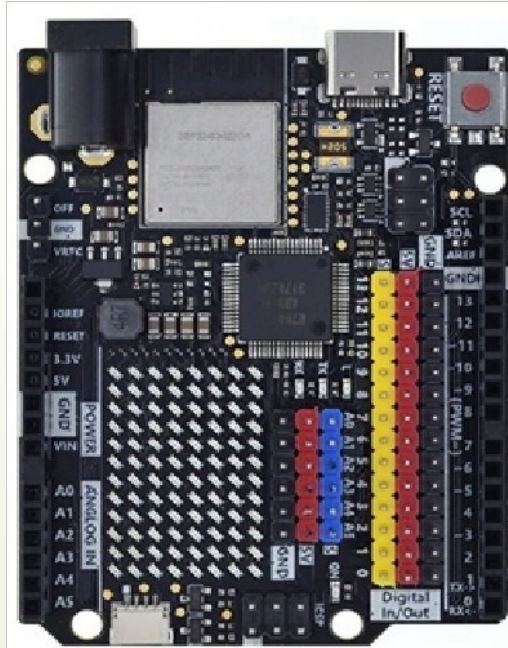


Arduino UNO+ESP8266 – HTTP POST - URL enkodiran



POST /update HTTP/1.1
Host: example.com
api_key=api&field1=value1
Content-Type: application/x-www-form-urlencoded

Arduino UNO+ESP8266 – HTTP POST - JSON object



HTTP POST

1 /update

Body JSON Object $\left\{ \begin{array}{l} \text{"api_key"} : \text{"API_KEY"}, \\ \text{"field1"} : \text{"30"} \end{array} \right\}$

HTTP Response

2 Status 200 (OK)



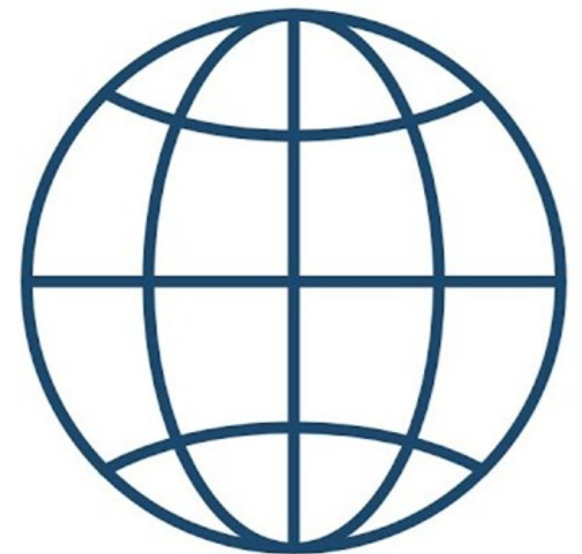
POST /update HTTP/1.1
Host: example.com
{api_key: "api", field1: value1}
Content-Type: application/json

Upisivanje podataka u ThingSpeak kanal

ESP8266 SEND DATA To WEBSITE



**ThingSpeak
WEBSITE**



Za vježbu

1. Upotrijebiti sensor za temperaturu i vlagu. Vrijednosti dobijene sa senzora slati na ThingsSpeak i prikazivati u dva odvojena dijagrama istog kanala.

(1)

2. Upotrijebiti koračni motor. Koračnom motoru zadavati sljedeće komande putem seriskog monitora: brzinu obrtanja kao kor/sec, smjer obratanja (lijevo, desno), kao i start i stop komande. Prije zadavanja komandi motor miruje. Kada se poslije STOP komande, zada START komanda, motor treba da nastavi pomjeranje brzinom i smjerom zadatim prije zadavanja STOP komande. U jenom dijagramu na ThingsSpeak kanalu prikazivati brzinu i smjer kretanja koračnog motora na sljedeći način:

- brzinu prikazivati vrijednošću po vertikalnoj osi dijagrama;
- za smjer LIJEVO, vrijednost po vertikalnoj osi dijagrama treba biti pozitivna;
- za smjer DESNO, vrijednost po vertikalnoj osi dijagrama treba biti negativna;
- kada motor miruje vrijednost po vertikalnoj osi dijagrama treba biti jednaka nuli;

(2-1)

3. Upotrijebiti servo motor. Poziciju servo motoru, odnosno ugao zakretanja, zadavati sa ThingsSpeak platforme, upotrebom TalkBack aplikacije.

U jednom dijagramu ThingsSpeak kanala prikazivati pozicije servo motora u vremenu, odnosno ugao zakretanja u rasponu od 0 do 180 stepeni.

(2-1)