

Microsoft Bot Frameworkin käyttö oppilaitoksessa



Ammattikorkeakoulututkinnon opinnäytetyö

Visamäki, tietojenkäsittely

kevät, 2017

Otto Laine

Tietojenkäsittelyn koulutusohjelma
Visamäki, Hämeenlinna

Tekijä	Otto Laine	Vuosi 2017
Työn nimi	Microsoft Bot Frameworkin käyttö oppilaitoksessa	
Työn ohjaaja	Tommi Saksa	

TIIVISTELMÄ

Opinnäytetyön idea tuli Hämeenlinnan ammattikorkeakoululta. Tutkimuksen tavoitteena oli selvittää, voitaisiinko Microsoft Bot Frameworkilla tehtyjä botteja käyttää oppilaitoksessa, joko oppilaitosta hyödyntävänä tuotteena tai opiskelijoille harjoitustehtävänä.

Teoriaosuudessa keskityttiin käymään läpi keskustelu-bottien historiaa ja Microsoft Bot Frameworkin toiminnallisuutta. Tutkimusmenetelmänä käytettiin aivorihtä, jolla kerättiin ideoita botin aiheen valintaan.

Käytännön osuudessa rakennettiin toimiva keskustelu-botti, joka kykenee käymään ohjatun keskustelun käyttäjän kanssa. Sen lataaminen dokumentoitiin Azure-pilvipalveluun. Tämän jälkeen botti ohjattiin Microsoft Bot Directoryyn.

Botti osoittautui toimivaksi ja sitä ja sen kaltaisia voitaisiin käyttää harjoituksina opiskelijoille.

Avainsanat tekoäly, prototyyppi, aivorihi, Microsoft Bot Framework, Azure

Sivut 24 sivua

Degree Programme in Business Information Technology
Visamäki, Hämeenlinna

Author	Otto Laine	Year 2017
Subject	Microsoft Bots usability at a University	
Supervisor	Tommi Saksa	

ABSTRACT

Idea for thesis came from Häme University of Applied Sciences. The study purpose of the study was to find out if bots created with Microsoft Bot Framework could be used at the university as a product or an assignment for the students.

The theory section covers the history of chat bots and how Microsoft Bot Framework works. The Research method used in this project was brain-storming which was used to gather ideas to create a bot.

In the practical part of the study, a bot was created and it is capable of simple guided conversations with the user. The bot's upload to Azure cloud service was documented. Afterwards the bot was added to Microsoft Bot Directory.

Bot was working as expected and it and similar ones could be used as a assignments for students.

Keywords artificial intelligence, prototype, brain storming, Microsoft Bot Framework, Azure

Pages 24 pages

Sanasto

Botti – ohjelma joka toimii käyttäjän hyväksi tarjoten jotain palvelua

C# - ohjelmointi kieli

Azure - Microsoftin tarjoama pilvipalvelu (internetin yli toimiva varastointi tila)

Framework – tukirakenne, kehys. Tässä tutkimuksessa kyseessä on Microsoftin tarjoama kehys bottien rakentamiseksi

SISÄLLYS

1	JOHDANTO.....	1
2	KESKUSTELU-BOTEISTA.....	2
2.1	Mikä on keskustelu botti?	2
2.2	Bottien historiaa.....	2
3	MICROSOFT BOT FRAMEWORK.....	6
3.1	Yleistä Frameworkistä	6
3.2	Bot Connector	7
3.3	Bottien viestintä	7
3.4	LUIS.....	8
3.5	Viestintäkanavat.....	10
3.6	Keskustelutietojen tallennus.....	11
3.7	Botin siirtäminen Azureen.....	11
4	AIVORIIHI	12
4.1	Opinnäytetyön aivoriihen tavoitteet	12
4.2	Aivoriihen eteneminen.....	12
5	BOTIN RAKENTAMINEN	15
5.1	Ensimmäinen yritys: lukujärjestys-botti.....	15
5.2	Ensimmäinen testi-botti.....	15
5.3	Create a Dog botti	16
5.3.1	Create a Dog Botin testaaminen	18
5.3.2	Botin Azureen laittaminen.....	19
5.3.3	Botin rekisteröiminen Microsoftin Bot Directoryyn.....	21
6	JOHTOPÄÄTÖKSIÄ.....	24
6.1	Botin rakentamisen haasteet	24
6.2	Botin rakentaminen opiskelijan toimesta tehtävänä.....	24
	LÄHTEET	25

1 JOHDANTO

Opinnäytetyön ideana on rakentaa prototyyppi keskustelu-botti ja miettiä voitaisiinko bottia käyttää oppilaitoskäytössä. Inspiraatio ideaan tuli suuresta mielenkiinnosta tekoälyn toimintaan ja mahdollisuuden tarjosi Hämeenlinnan ammattikorkeakoulu. HAMK halusi, että selvitetään, voitaisiinko botteja käyttää oppilaitoksen toimien tukena, tai tehtävinä joilla opiskelijat tutustuisivat miten botteja rakennetaan ja mihin kaikkeen niitä voidaan käyttää. Idea prototyyppiä varten saadaan järjestämällä aivoriihi, jossa kerätään ideoita. Näistä ideoista valitaan mielenkiintoisin ja kehityskelpoisin. Tästä ideasta pyritään rakentaa toimiva prototyyppi. Teoriaosuudessa käydään läpi keskustelu-bottien historiaa ja miten bottien rakentaminen Frameworkilla toimii teoriassa. Teoriaosuuden lopussa käydään myös lyhyesti lävitse, kuinka aivoriihi järjestetään minkä jälkeen siirytään aivoriihen käytännön osuuden selostukseen ja aiheen valintaan. Käytännön osuudessa käydään läpi prototyypin suunnittelu ja rakentaminen sekä miten se ladataan Azure-pilvipalveluun ja sen jälkeen aktivoidaan Microsoftin Bot Directoryssa. Lopuksi vielä käydään läpi johtopäätöksiä, joissa pohditaan mitä haasteita projektissa oli ja voitaisiinko bottia käyttää oppilaitoskäytössä. Tavoitteena on selvittää, miten botteja rakennetaan ja voitaisiinko niitä käyttää harjoituksina opiskelussa taikka muuten oppilaitoksen hyötykäytössä.

2 KESKUSTELU-BOTEISTA

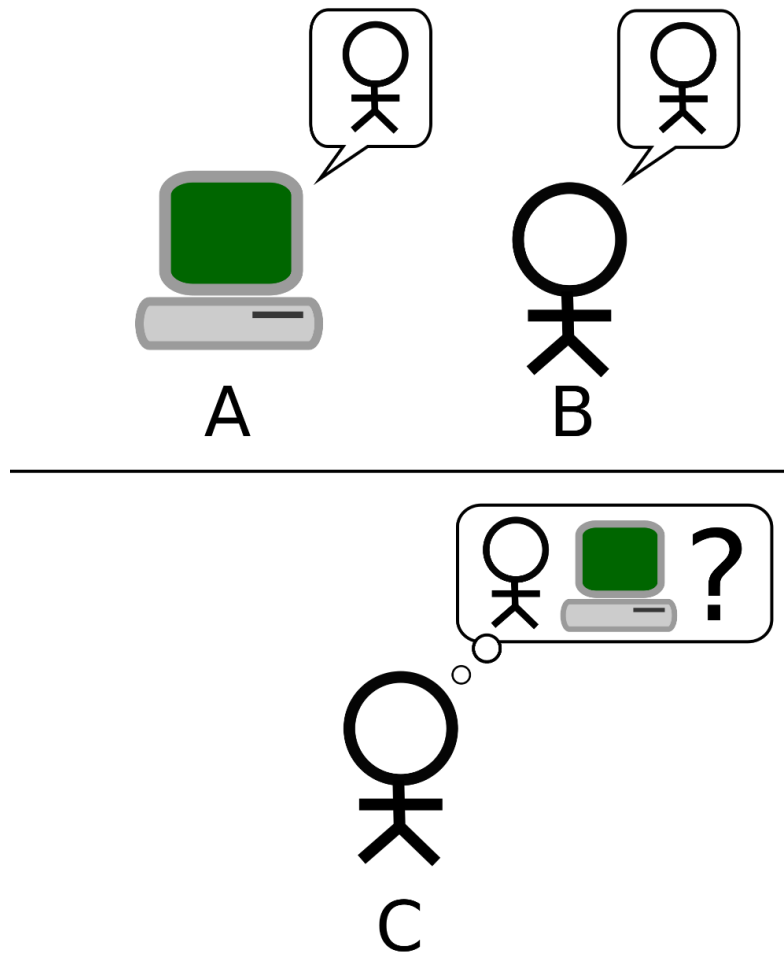
2.1 Mikä on keskustelu botti?

Keskustelu-botit ovat botteja, jotka pystyvät käymään keskustelua ihmisen kanssa. Keskustelun ei tarvitse olla kovin syvällistä eikä se pysy kovin pitkään selkokielisenä, sillä monet botit menevät nopeasti sekaisin monimutkaisemmista kysymyksistä ja eivät siedä kielellisiä virheitä. Yleisimpiä käyttötarkoituksia keskustelu-boteille ovat mm. varaupalvelut, tiedustelupalvelut ja keskustelupalvelut. Botit eivät ole oikeasti älyllisiä vaan ammentavat tietonsa sisäisestä hakukoneesta ja ymmärtävät kieltä LUIS-protokollan taikka vastaavien systeemien avulla. Kuuluisimpia keskustelubotteja ovat Applen Siri ja Microsoftin Cortana.

2.2 Bottien historiaa

Alan Turing kehitti vuonna 1950 Turingin testin, jonka ideana oli selvittää, voiko ihmispelaaja havaita kumpi kahdesta vastaajasta on botti. Ihmispelaaja esitti kysymyksen, johon hän saa kaksi vastausta. Vastauksesta pelaajan piti päätellä, kumpi vastaajista oli botti. Systemi ei tietenkään ollut täydellinen, sillä enemmän tai myöhemmin botti vastasi jotenkin hassusti taikka muuten epäkäytännöllisesti, paljastaen itsensä. Turingin testi kuitenkin osoitti, että ihmisillä on syvä kiinnostus pystyä keskustelemaan myös tietokoneitten, ei pelkästään ihmisten kanssa. (Bayerque 2016; Wotawa n.d.)

Turingin testiä on kritisoitu, sillä testi ei muun muassa ota huomioon ihmisten välillä irrationaalista käytöstä taikka outoja vastauksia. (Wotawa n.d.)



Kuva 1. Turing Testi (Hugo Férée, Wikimedia Commons)

Yllä olevassa kuvassa (Kuva 1) näkyy selitys miten Turingin testi toimii. Ideana on että käyttäjä esittää toteamuksen taikka kysymyksen johon tulee kaksi vastausta. Käyttäjä ei tiedä kumpi vastaajista on botti ja hänen täytyy arvata vastausten perusteella kumpi vastauksista todennäköisemmin on ihmisen antama. (Férée 2016.)

Elizan kehitti Joseph Weizenbaum vuonna 1966 ja se oli maailman ensimmäinen keskustelemaan kykenevä botti. Eliza oli suunniteltu toimimaan kuin terapeutti, sillä se vastasi kysymyksiin tunteellisesti käyttäen hyväksien avainsanoja ja yleensä vastaten kysymyksellä, miten henkilö ajatteli tilanteestaan. Eliza kehitettiin alun perin parodioimaan terapeuttien tyyppillistä kysymysasettelua ja sitä, kuinka pinnallista se on, mutta kehittäjä yllättyikin kun ihmiset muodostivat vahvoja kiintymystunteita bottia kohtaan koska kokivat sen oikeasti auttavan. (Godara 2016.)

```

Talk to Eliza

* Hi
> How are you today.. What would you like to discuss?
* I feel lonely
> Do you enjoy feeling lonely?
* no
> Are you saying no just to be negative?
* no, i just feel sad because i am lonely
> How long have you been lonely?
* forever
> OK... "no, you just feel sad because you are
lonely". Tell me more.

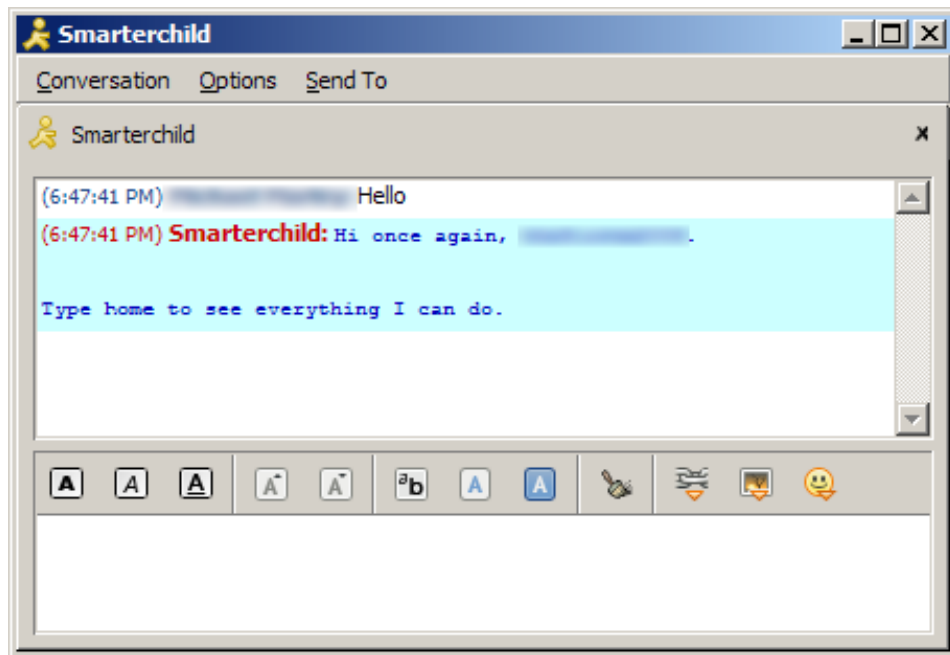
Input: 

```

Kuva 2. Eliza

Yllä olevassa kuvassa (Kuva 2) näkyy lyhyt keskustelu, jonka kävin Elizan kanssa. Eliza oli ensimmäinen botti, jonka kanssa käyttäjät muodostivat tunnesiteitä, vaikka botti vastasikin vain käyttäen hyvin yksinkertaista lauseitten kysymykseksi kääntämisen logiikkaa. (Godara 2016.)

Iso edistysaskel keskustelu-boteissa tapahtui kun start-up-yritys Active-Buddy julkaisi botin nimeltä SmarterChild, jolla oli kaksi tärkeää kykyä: se kykeni muistamaan aikaisemmat keskustelut ja se pystyi ymmärtämään ihmispuhetta, ei vain lukemaan avainsanoja. SmarterChild julkaistiin vuonna 2000 ja se toimi AIM-keskusteluohjelman kautta joka oli tuohon aikaan suosittu keskusteluohjelma. SmarterChildin kanssa pysty käymään tavallista keskustelua monista aiheista, kuten urheilusta, säästä ja vaikka elokuvanäytösten ajoista. SmarterChild oli myös erittäin suosittu koska se tarjosi nopeamman tavan saada informaatiota tuolloin verrattuna Googlen ja Yahoo!:n palveluun. (Rodrigues 2016.)



Kuva 3. SmarterChild (Wikimedia Commons)

Yllä olevassa kuvassa (kuva 3) ennestään tuttu käyttäjä aloittaa uuden keskustelun SmarterChildin kanssa. Kyky muistaa käyttäjän kanssa aikaisemmin käydyt keskustelut oli ominaisuus joka oli uusi keskustelu-boteille. SmarterChild oli ensimmäinen joka käytti sitä. (Rodrigues 2016.)

SmarterChild ja muut sitä edeltäneet botit kuten Alice ja Mitsuku loivat perustan nykypäivän hyvinkin edistyneille keskustelu-boteille. Näistä tunnetuimpia esimerkkejä ovat Applen Siri ja Microsoftin Cortana.

3 MICROSOFT BOT FRAMEWORK

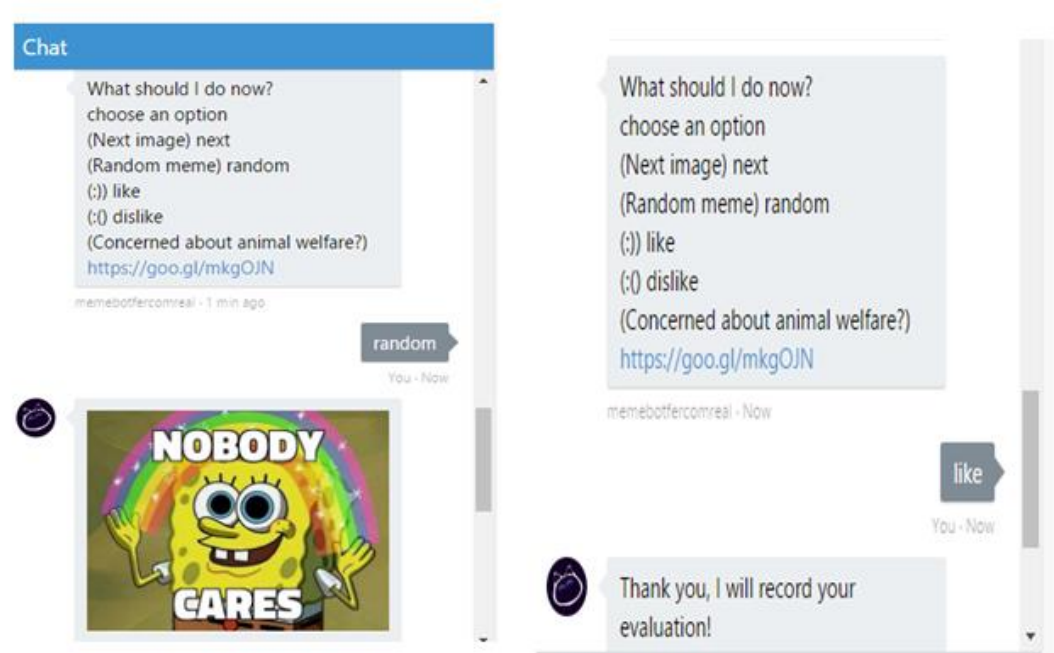
3.1 Yleistä Frameworkistä

Microsoft Bot Framework on tarkoitettu luomaan botteja, jotka kykenevät joko tarjoamaan hyödyllistä informaatiota käyttäjälle taikka käymään keskustelua käyttäjän kanssa, mieluiten käyttäjän omalla äidinkielellä ja missä tahansa ympäristössä, oli se kännykällä taikka kotikoneella. Framework tarjoaa mahdollisuuden luoda botteja käyttäen joko C#-koodauskieltä, Javascriptiä taikka REST-apia joka perustuu jsoniin. Ideana Microsoftilla oli luoda palvelu, jolla on mahdollisimman helppo tehdä botteja, joita sitten voidaan käyttää mm. Facebookissa taikka Skypessä. Kaikki botit, jotka Frameworkia käyttäen luodaan, on siis tarkoitettu toimimaan netin kautta, ei lokaalisti.

Botteja voi olla kahdenlaisia: toiset botit ovat hyvinkin avoimia ja käyttävät LUIS-protokollaa ymmärtääkseen puhuttua kieltä. Yleisemmät ja ehkä tunnetummat botit perustuvat ns. keskustelupuurakenteeseen, jossa botin kanssa voidaan keskustella järkevästi käyttämällä vain tiettyjä avainsanoja. Kuten vaikkapa parturivaraus-botti ei vastaa, ellei sano ensin päivämäärää, milloin haluat varata ajan.

Bot Frameworkin SDK tarjoaa myös liudan ominaisuuksia, joita voi liittää bottiisi. Aikaisemmin mainittu LUIS-protokolla mahdollistaa kielen ymmärtämisen ja sen lisäksi voi liittää Microsoftin Cortanan puhe ääneksi botillesi ja Bingin hakukoneeksi niin että botti voi tarvittaessa hakea reaalitajassa tietoa netin kautta.

Microsoft myös tarjoaa mahdollisuuden julkaista botti, mutta tämä ei ole pakollista, ellei halua kytkeä bottia Skypen koska silloin botti tarvitsee ID:n. (Microsoft 2016; Bisson 2016.)



Kuva 4. Memebotin kanssa käyty keskustelu

Yllä on kaksi kuvaa Frameworkilla tehdyn botin kanssa käydystä keskustelusta. Tämä botti toimii keskustelupuu-periaatteella, eli se reagoi vain tiettyihin kommentteihin ja muuten ei reagoi käyttäjän kommentteihin. (Kuvat 4 ja 5.)

3.2 Bot Connector

Palvelu joka on tärkeä bottien luomisessa on Bot Connector, jota tarvitaan että botti voi saada yhteyden käyttäjään. Tärkeä tehtävä jonka Connector myös tekee on sen kyky muuntaa tietoa käytettyyn kanavaan sopivaksi (esim. kuva johon on sisällytetty toimintanäppäimet ”vahvista” ja ”älä vahvista” muunnetaan kuvaksi, jonka kanssa tulee teksti ”vastaa kirjoittamalla vahvista taikka älä vahvista”).

Tämän projektin kannalta helpottavana tekijänä Microsoft tarjoaa C#-kieltä käyttävän Botin rakennuspaketin joka mallintaa Connetoria. Lisäksi rakennuspaketti tarjoaa valmiita pohjia ja dialogeja bottien rakentamista varten. (Microsoft 2016)

3.3 Bottien viestintä

Yleensä botin ja ihmisen välinen keskustelu aloitetaan ihmisen toimesta. Tällaisessa tilanteessa käyttäjä lähettää message-tyyppisen viestin, jolloin

botin täytyy vastata täyttäen tietyt ehdot. Botin pitää määrittää mikä keskustelu on kyseessä, keneltä alkuperäinen viesti tulee, mistä viesti on tullut eli käyttäjän sijainti, kenelle viesti menee, vahvistaa viestin saajan ID ja viimeiseksi vielä tähdentää viestin tyyppiä message. Lisäksi jos botti vastaa tekstinä voidaan viestiin määrittää text-tyyppi. Kuvia tai videoita tai vastaavia sisältäviä viestejä varten pitää määrittää attachment- taikka attachmentLayout-arvo.

Botti harvemmin aloittaa keskustelua Yleensä näissä tilanteissa kyseessä on ilmoitus, jolla huomautetaan jostakin käyttäjää koskevasta asiasta, vaikka harrastuksesta taikka käyttäjän intressejä koskevasta uutisesta. Tämän tehdäkseen botilla pitää olla ennalta annettua informaatiota käyttäjästä. Lähettäkseen viestin botin pitää antaa ensin oma informaatiossa ja sitten halutun käyttäjän informaatio, sen jälkeen viesti etenee saman lailla kuin vastaamisessa. (Microsoft 2016.)

3.4 LUIS

LUIS tunnetaan myös Language Understanding Intelligent Servicenä. Se on Microsoftin kehittämä palvelu, joka kykenee tulkitsemaan ihmisen kieltä ja etsimään siitä tarkoitusta.

```
{
  "query": "turn the lights off",
  "topScoringIntent": {
    "intent": "TurnAllOff",
    "score": 0.9751273,
    "actions": [
      {
        "triggered": true,
        "name": "TurnAllOff",
        "parameters": []
      }
    ]
  },
  "entities": [],
  "dialog": {
    "contextId": "a6d06cfb-866a-4271-8ee6-60b91b54040c",
    "status": "Finished",
    "prompt": null, "parameterName": null
  }
}
```

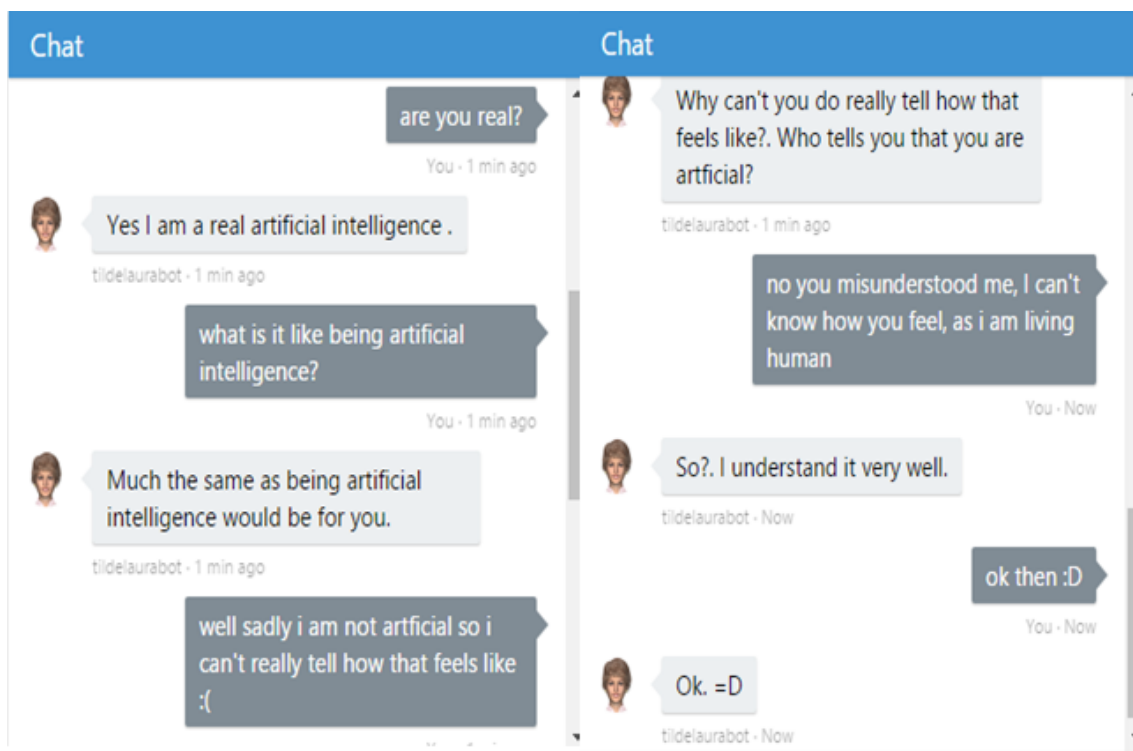
Yllä on Microsoftin käyttämä esimerkki, jossa käyttäjä pyytää tekoälyä sammuttamaan kaikki valot ja alla on esimerkki, jossa tekoäly käynnistää vain työpöydän valon.

```
{
  "query": "switch on the desk lamp",
  "topScoringIntent": {"intent": "TurnOn", "score": 0.9920862,
  "actions": [
    {
      "triggered": true, "name": "TurnOn", "parameters": [
        {
          "name": "Light", "required": true, "value": [
            { "entity": "desk", "type": "Light" }
          ]
        },
        { "name": "Color", "required": false, "value": null }
      ]
    }
  ]
},
  "entities": [
    { "entity": "desk", "type": "Light", "startIndex": 14, "endIndex": 17,
      "score": 0.998846948,
      "resolution": {
        "date": null, "time": null }
    }
  ],
  "dialog": { "contextId": "233fcd33-3c01-42b5-a8d8-82843d088632",
    "status": "Finished", "prompt": null, "parameterName": null }
}
```

Kuten esimerkeistä havaitaan, LUIS kykenee huomaamaan jopa vain muuttaman sanan poikkeaman.

LUIS itsessään on tarkoitettu kehittämään rakenteita joilla botit ymmärtävät kieltä ja pystyvät aktiivisesti oppimaan ja ymmärtämään kieliä paremmin käyttäen niin kutsuttua itse-oppimista. Itse-oppimisen ideana on että botit poimii kokemuksistaan ihmisten kanssa uusia sanoja valikoimaansa ja voi joko käyttää taikka ymmärtää niitä.

Tällä hetkellä LUIS on tuettu pääasiassa englanniksi, mutta se kykenee myös ymmärtämään mm. saksaa, ranskaa ja kiinaa. (Microsoft 2016; Bisson 2016.)



Kuva 5. Laura botin kanssa käyty keskustelu

Yllä olevassa kuvissa (Kuvat 6 ja 7) jotka otin yhden LUIS protokollaa käyttävän botin kanssa käydystä keskustelusta. Hiukan yllättäen vastauksista tuli paljon mieleen bottien historiaosuudessa mainitsemani ensimmäinen tekoäly, Eliza. Monet Lauran vastauksista tuntuivat hyvin samankaltaisilta, eli minun lauseitteni kääntämistä kysymyksiksi. Botista kuitenkin huomaa selvästikin että se ymmärtää mitä minä sanon, ainakin tiettyyn rajaan saakka, sillä selvästikkin monimutkaisemmat lauseet menevät yli sen käsittelykyvyn.

3.5 Viestintäkanavat

Viestien lähettämisessä ja mitä asioita viesti sisältää pitää aina muistaa tarkistaa käytetyn kanavan (esim. Skype) dokumentaatiota niin että bottilla ei ole vaikeuksia lähettää viestejä.

Eräät kanavat vaativat tavallisesta poikkeavia arvoja, esimerkiksi Office 365-sähköpostin lähettämiseen tarvitaan uudet arvot `htmlBody`, `importance` ja `subject`. `Html` sisältää viestin sisäkkö, `importance` määrittää viestin tärkeyden ja `subjectiin` tulee viestin otsikko. (Microsoft 2016.)

3.6 Keskustelutietojen tallennus

Jotta Botti voi aloittaa viestiketjun se tarvitsee aikaisemmin tallennettua informaatiota käyttäjästä. Tämä data voidaan tallentaa kolmella eri tavalla, jotka kaikki on tarkoitettu eri tilanteisiin. Käyttäjätietojen tallentaminen kanavalta joka ei ole vain sitä keskustelua varten tallentaa User Data Storeen. Käyttäjätietojen keskustelua varten olevalta kanavalta tallennetaan Private User Data Storeen ja jos haluaa tallentaa yleistä tietoa keskustelusta ilman tarkempia käyttäjäkohtaisia tietoja, tallennetaan se Conversation Storeen. (Microsoft 2016.)

3.7 Botin siirtäminen Azureen

Botti kannattaa ladata valmistumisen jälkeen Microsoftin Azure-pilvipalveluun, koska silloin bottia varten ei tarvitse pitää omia servereitä. Lisäksi Azuren kautta on helppo määrittää millä alustoilla botti toimii (esim. Facebook).

Tämän voi toteuttaa kolmella eri tavalla (omasta git-kansiosta, Githubista taikka Visual-studiosta). Tässä projektissa keskitytään Visual Studion tapaan koska botti toteutetaan sillä.

Aikaisemmin projektissa on jo ladattu Bot Builder SDK, joten ei tarvitse tehdä muuta kuin asentaa puuttuvat tiedostot, jotka eivät tule oletuksena mukana. Tämän jälkeen `Web.config`-tiedostoon lisätään `botframework_appid`- ja `botframework_appsecret`-arvot (nimetään botin rekisteröimisen jälkeen). Kun nämä vaiheet on toteutettu ei tarvitse tehdä kuin solution explorerista valita julkaise, kirjautua sisään Azureen ja tehdä uusi Web-aplikaatio.

On myös hyvä käytäntö testata, että botti toimii Azuressa käyttämällä Bot Framework Emulaattoria, johon ei tarvitse kuin kopioida botin url-osoite jonka saa julkaisun jälkeen. (Microsoft 2016; Bisson 2016)

4 AIVORIIHI

Onnistunut aivoriihi aloitetaan sääntöjen sopimisella. Yleisimmät käytännöt ovat, että kaikki antavat ehdotuksia vuoron perään ja jos ei keksi ideaa niin voi vain sanoa ohi. Tässä vaiheessa on tarkoitus kerätä ideoita, joten mielikuvituksellisimmat ja hulluimmatkin ideat hyväksytään. Aivoriihen vetäjän tehtävänä on huolehtia, että keskustelu jatkuu ja hän myös taltioi kaikki ryhmän jäsenten ideat näkyville.

Kun idea virta alkaa tyrehtyä, aloitetaan pohtimaan mitkä ideoista ovat oikeasti toteutettavissa, mutta jos ideoita vielä tulee, ne otetaan vastaan. Ideana on antaa rakentavaa kritiikkiä ideoista ja pohtia voisiko joitakin ideoista yhdistää. Varsinainen aiheen tai ratkaisun valinta ei pakosti tapahdu itse aivoriihessä, mutta jos se toteutetaan sen yhteydessä, se voidaan esim. toteuttaa antamalla kaikkien riihen jäsenten merkata plusmerkki mielestään hyvien ideoitten viereen. (Lavonen, Meisalo n.d.)

4.1 Opinnäytetyön aivoriihen tavoitteet

Aivoriihen tavoitteena on kerätä ideoita vapaasti ja pohtia mitkä niistä olisivat toteutettavissa. Ideana olisi saada kerättyä iso lista ideoita, joista sitten karsitaan muutama kehitys kelpoisin, minkä jälkeen itse päätän yhden niistä, josta pyrin kehittämään toimivan prototyypin.

Toisena tavoitteena, on myös kokea, miten aivoriihen toteuttaminen toimii tiedonkeruumenetelmänä ja mitä hyötyjä ja mahdollisia haittoja siitä on.

4.2 Aivoriihen eteneminen

Aivoriihi aloitettiin torstai aamuna 26.1.2017 kello 8.45 aivoriihen. Paikalla oli kaksi opettajaa ja yksi toinen opiskelija. Tavoitteena oli kerätä ideoita ja miettiä mitkä niistä olisivat toteutuskelpoisimpia projektiani varten. Ensiksi käytiin läpi säännöt, eli ideoita ei kritisoidaisi ennen kuin kaikki mahdolliset ajatukset oli saatu ylös. Tämän jälkeen esitettiin läsnäolijoille lyhyt PowerPoint-esitelmä Microsoft Bot Frameworkin toiminnasta, jonka jälkeen tutkittiin Microsoftin virallista dokumentointia, jotta kaikki saivat ainakin perusymmärryksen, miten botit toimivat ja mihin niitä voidaan käyttää.

Tämän jälkeen alettiin kerätä ideoita ja keskustelu etenikin mainiosti ja ideoita alkoi kertyä hyvin runsaasti. Kun läsnäolijoitten mielestä oli saatu kerättyä kaikki ideat mitä mieleen tuli, alettiin pohtia mitkä ideoista olisi-

vat toteutuskelpoisia ja voisiko joitakin ideoista kenties yhdistää. Aivoriihessä päädyttiin 5 suhteellisen yksinkertaiseen botti-ideaan, jotka kattavat paljon erilaisia projekteja. Kaksi mainitsemisen arvoista ehdotusta joita ei valittu: Ensimmäinen on ohjelmointia koskeva botti, sillä sen ideana on tehdä botti ja miettiä olisiko tällaisen botin rakentaminen sopiva tehtävä opiskelijoille. Toinen idea koski bottia joka toimisi koulun mainonnan tukena, sen ideana oli luoda botti, joka kysyy käyttäjältä kysymyksiä ja niiden perusteella suosittelisi koulutuslinjaa. Ideana ei siis ollut tehdä mitään massiivista bottia joka antaisi tarkan arvion sopivasta linjasta, sen sijaan sillä olisi tarkoitus näyttää minkälaisia tuotoksia HAM-Kissa voidaan tehdä.

Kokonaisuutena aivoriihi oli erittäin tehokkaaksi tiedonkeruumenetelmäksi ja ideoita tuli erittäin runsaasti. Mahdollisia ongelmia voisi syntyä aivoriihessä, jos ryhmän jäsenet eivät ymmärtäisi aihetta, mutta tässä ryhmässä sitä ongelmaa ei syntynyt.

-
- [Ohjelmointi](#)
 - Esim. valitse itsellesi sopiva koira rotu
 - [Lukujärjestysbotti](#)
 - [Ruokalistabotti](#)
 - Kaksi ylempää yhdessä
 - Opettaja [faq](#) assistentti
 - Esim. Lähdeviittaukset
 - Toiminnanohjausjärjestelmä (tilaus prosessin tukena)
 - [Etsi vapaa tila \(osa lukujärjestysbottia?\)](#)
 - Ohjelmoinnin opetusaiheet
 - SDK? Käytettävissä
 - Manuaalisesti vs. SDK
 - Moodlen viikon tehtävät
 - Moodlen viimeisin arvosteltu tehtävä
 - Moodlesta viimeisin viesti opettajalta

Kuva 6. Aivoriihen tuloksia

- [Markkinointi \(uraneuvonta botti – suosittelee vastausten perusteella koulutusohjelmaa\)](#)
- [Sääbotti](#)
- [Junaaikataulu botti](#)
- Hanke [botti](#)
 - Mistä saa rahaa, millä kriteereillä, milloin, [kirjottamis](#) apu
- [Vaihtopaikka etsintä botti](#) (kysymys pohjainen)

Kuva 7. Aivoriihen tuloksia

Yllä on kaksi kuvaa (kuvat 8 ja 9), jotka ovat PowerPoint esityksen, jossa selitin bottien toimintaa kahdelta viimeiseltä sivulta. Kaikki näkivät ideat tietokoneeseen liitetyn videotykin projektorilta. Kuvissa punaisella merkityt ideat ovat ideoita, jotka valittiin mahdollista jatkokehitystä varten (kuvat 5 ja 6).

5 BOTIN RAKENTAMINEN

Botin rakentamista varten ladattiin Microsoftin suosittelemat sovellukset, tässä tapauksessa Microsoft Visual Studio 2015 Community version sekä Bot Framework Simulatorin jolla botteja voidaan testata ennen kuin ne laitetaan Azureen. Lisäksi Visual Studio piti päivittää aivan viimeisimpiin päivityksiinsä asentamisen jälkeen.

5.1 Ensimmäinen yritys: lukujärjestys-botti

Ensimmäinen oikea botti jota rakennettiin oli lukujärjestys-botti, joka osoittautui liian suureksi haasteeksi. Koulun avoin data systeemi toimi mainiosti, mutta Visual Studiosta c# :lla yhteyden saaminen sivulle osoit-tautui liian suureksi haasteeksi, joten päädyttiin vaihtamaan toiseen aivo-riihessä esille tulleetseen ideaan. Alla kuvana (kuva 8) mitä saatiin ai-kaiseksi lukujärjestys-botista.

```
public async Task<HttpResponseBody> Post([FromBody]Activity activity)
{
    if (activity.Type == ActivityTypes.Message)
    {
        using (var client = new HttpClient())
        {
            var info = activity.Text.Split(',');
            var values = new Dictionary<string, string>
            {
                { "startDate", "2017-02-01T00:00" },
                { "endDate", "2017-02-04T00:00" },
                { "studentGroup", "[TRTKNU15A3]" },
            };

            var content = new FormUrlEncodedContent(values);

            var responsePost = await client.PostAsync("https://opendata.hamk.fi:8443/r1/reservation/search", content);

            var responseString = await responsePost.Content.ReadAsStringAsync();
        }
    }
}
```

Kuva 8. Lukujärjestys-botin rauniot

5.2 Ensimmäinen testi-botti

Botti-kokeilu aloitettiin tekemällä valmiina SDK:n kanssa tulevan Echo-botti joka hyvin simppelellä tavalla vain toistaa mitä käyttäjä kirjoittaa ja laskee kirjainten määrän. Alla on pätkä koodia joka hoitaa käyttäjän kirjoituksen toistamisen ja sen pituuden laskemisen

```
Activity reply = activity.CreateReply($"You sent {activity.Text}
which was {length} characters");
```

Valmis koodi sitten pyöritetään mikä avaa näkymän, joka näkyy alla olevassa kuvassa. Siihen voisi halutessaan laittaa omaa tekstiään mutta se ei ole tässä vaiheessa tarpeellista.

Bot_Application

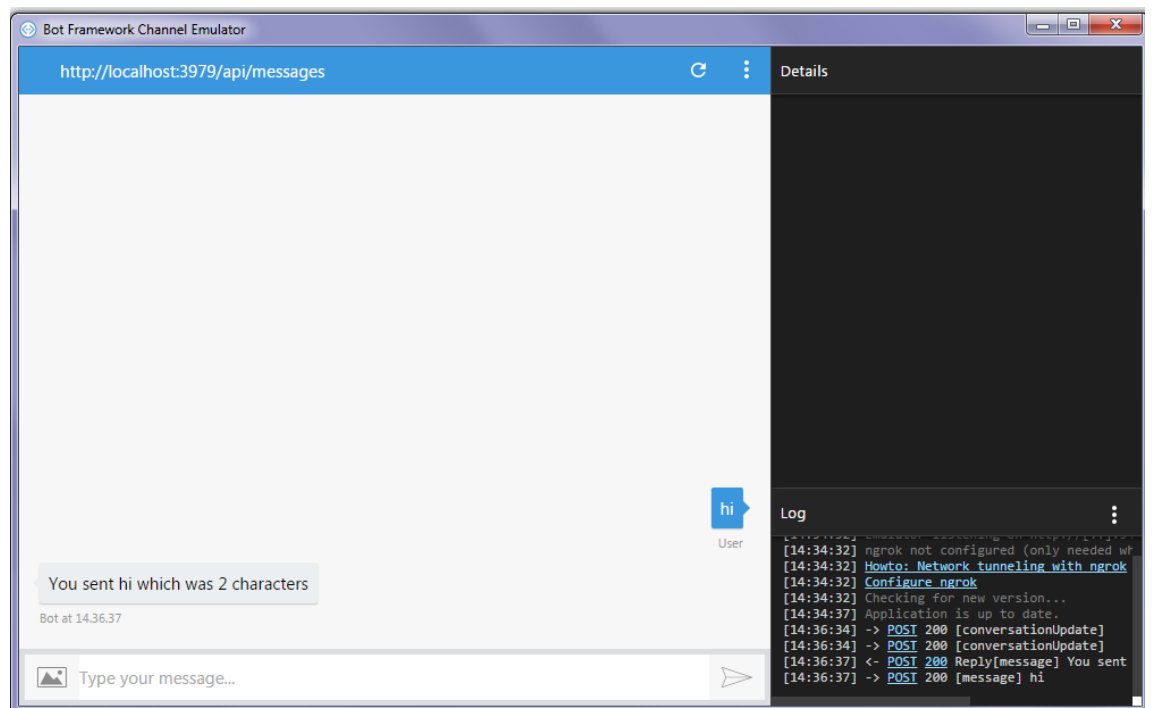
Describe your bot here and your terms of use etc.

Visit [Bot Framework](#) to register your bot. When you register it, remember to set your bot's endpoint to

`https://your_bots_hostname/api/messages`

Kuva 1. Testikoodi pyöritettynä

Pyöritetystä koodista haetaan URL-osoite, joka sitten syötetään Bot Framework emulaattoriin, jolla voidaan testata botin toimivuutta, kuten alla olevassa kuvassa (kuva 9) näkyy.



Kuva 9. Botti Emulaattori tosi toimissa

5.3 Create a Dog botti

Dog-botin ideana on toteuttaa yksinkertainen harjoitus botti jonka kuka tahansa opiskelija voisi tehdä. Se perustuu Microsoftin valmiiseen pohjaan, joten työskentely on nopeaa ja tehokasta. Botin toiminta perustuu FormFlowin käyttöön.

`using Microsoft.Bot.Builder.FormFlow;`

FormFlow tarjoaa mahdollisuuden luoda tehokkaasti ja sujuvasti eteneviä keskusteluita tarjoten samalla apua, jos käyttäjä niin haluaa.

Botille on määritetty useita eri arvoja, joista käyttäjä valitsee haluamansa ominaisuudet. Suurin osa ominaisuuksista on suunniteltu niin että käyttäjä valitsee yhden kahdesta tai useammasta vaihtoehdosta (esim. sukupuoli). Viimeinen vaihtoehto eli tempujen valinta on moni valinta eli käyttäjä voi halutessaan valita enemmän kuin yhden vaihtoehdon. Alla on koodi, jossa näkyvät kaikki mahdolliset vaihtoehdot joita käyttäjä voi valita varsinaisen keskustelun aikana.

```
namespace Bot_Application
{
    public enum DogOptions
    {
        LabradorRetriever, GoldenRetriever, GermanShepherd, Bulldog,
        Beagle, Poodle,
        Boxer, YorkshireTerrier, Rottweiler, Chihuahua, Dachshund,
        Husky, Dalmatian,
        ShibaInu, Corgi
    };
    public enum SexOptions { Female, Male };
    public enum AttitudeOptions { Playful, Solitary, Guard, Calm,
    Helpful };
    public enum AgeOptions { Puppy, Adult };
    public enum FurColorOptions
    {
        Black, Brown, White, Mixed, Hairless
    };
    public enum TrickOptions
    {
        PlayDead, ShakeHand, Fetch, Sit,
        Roll, Kiss, Bark, Spin, Stand, Hug
    };
    [Serializable]
    public class DogOrder
    {
        public DogOptions? Dog;
        public SexOptions? Sex;
        public AttitudeOptions? Attitude;
        public AgeOptions? Age;
        public FurColorOptions? FurColor;
        public List<TrickOptions> Trick;
    };
}
```

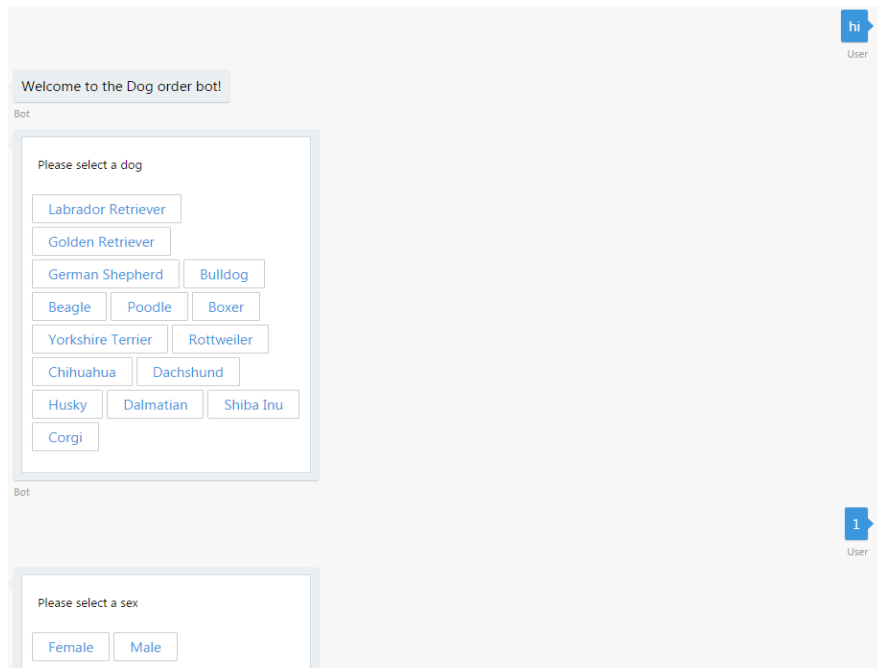
Viimeisenä asiana koodissa luodaan varsinainen Form, joka näkyy alla olevana koodinpätkänä. Tämä muodostaa kyselyn jonka käyttäjä sitten näkee aloittaessaan kyselyyn vastaamisen. Tämä on myös kohta johon MessageController ottaa yhteyden käyttäjän aloitettua keskustelun botin kanssa.

```
public static IForm<DogOrder> BuildForm()
{
    return new FormBuilder<DogOrder>()
        .Message("Welcome to the Dog order bot!")
        .Build();
}
```

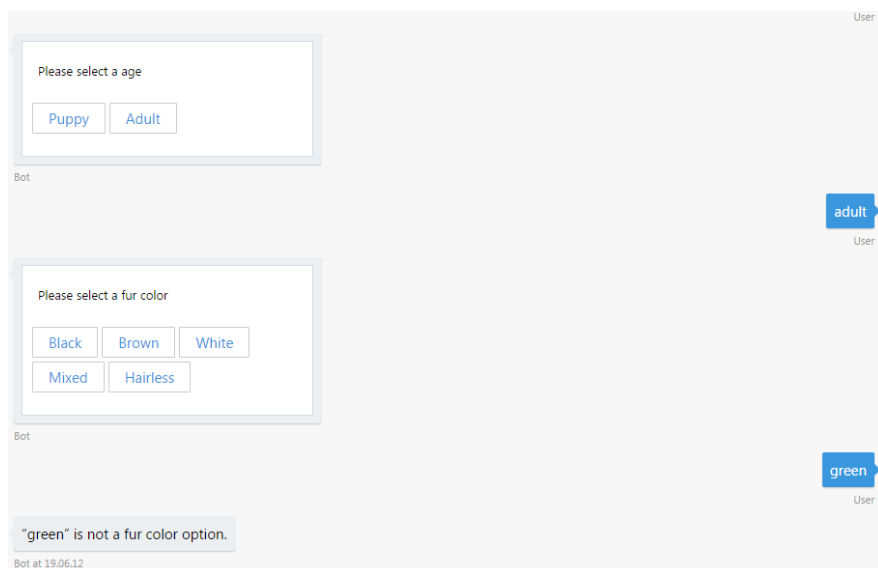
}

5.3.1 Create a Dog Botin testaaminen

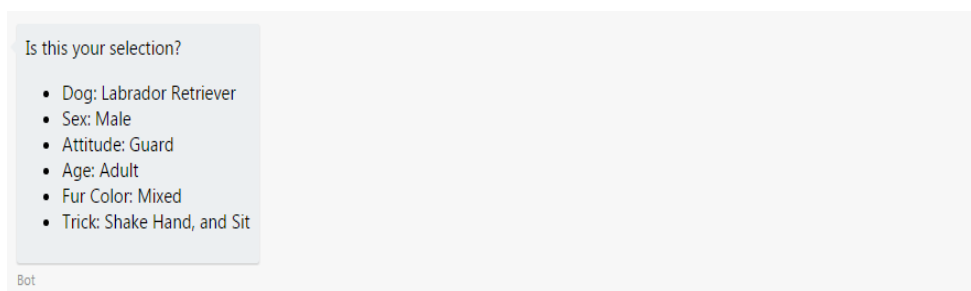
Kuten aikaisemmin Echo-botin kanssa, ensin testattiin Bot Framework Simulatorilla botin toimivuutta alla olevissa kuvissa (Kuvat 10, 11 ja 12).



Kuva 10. Create A Dog botin testaamista



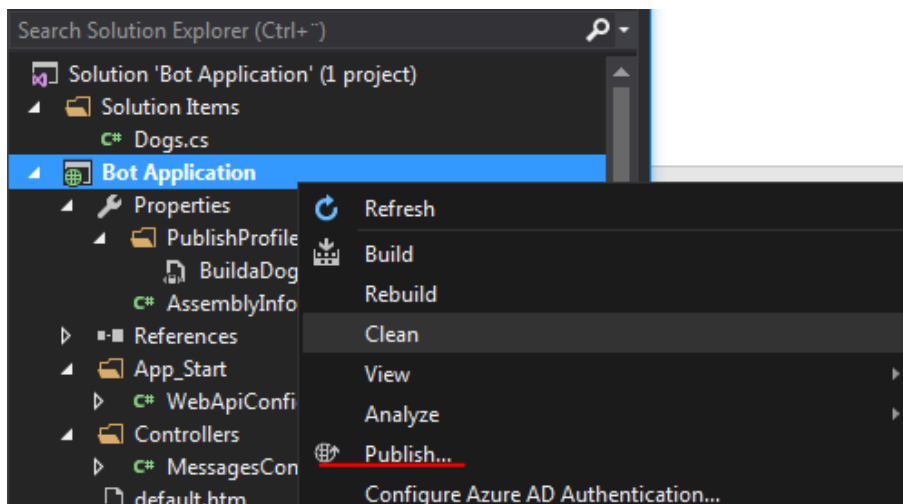
Kuva 11. Create a Dog botin testaamista



Kuva 12. Create a Dog botin testaamisen tulokset

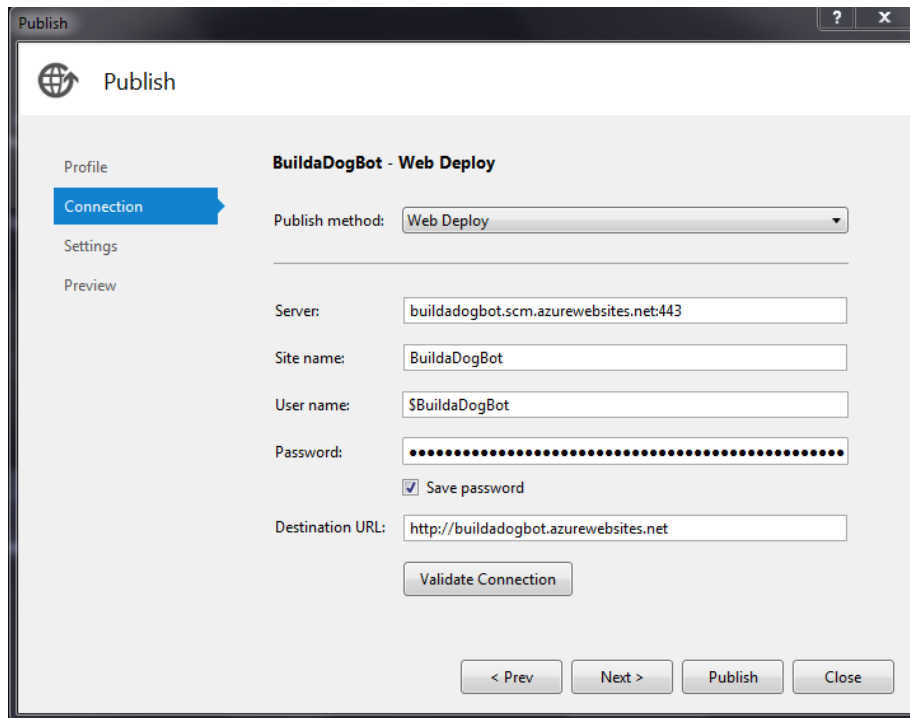
5.3.2 Botin Azureen laittaminen

Testauksen jälkeen aloitettiin botin julkaiseminen Azureen. Alla olevassa kuvassa (kuva 13) näkyy punaisella alleviivattuna Publish-ominaisuus, jonka kautta Azureen julkaisu toteutetaan.



Kuva 13. Visual Studiosta Publishaaminen

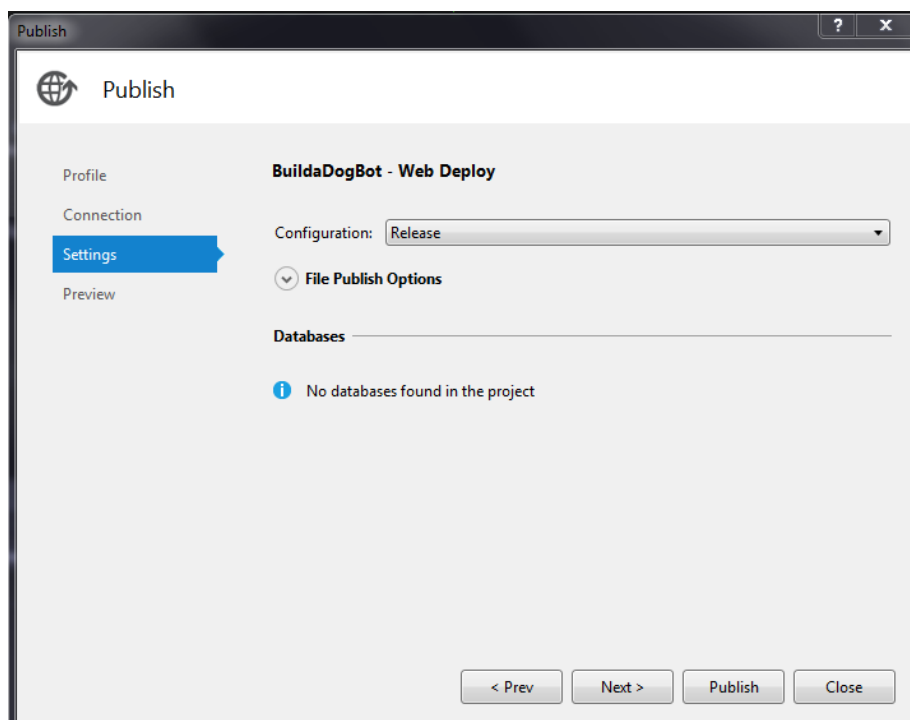
Tämän jälkeen Visual Studio avaa Install Wizardin joka ohjaa käyttäjän julkaisuprosessin läpi. Kannattaa huomioida, että Azureen julkaiseminen ei ole mahdollista ilman Microsoftin Azuren tilaamista kausimaksulla. Projektissa käytettiin ilmainen kokeilu-aika tämän projektin toteuttamiseen, mutta se ei pakosti ole kaikilla vaihtoehto, sillä tilaaminen vaatii luottokortin rekisteröitymistä varten. Prosessi on hyvin suoraviivainen ja ainut asia joka pitää muistaa muokata on julkaisutavan muuttaminen Applicationista Web Applicationiksi. Alla olevissa kuvissa (kuvat 14 ja 15) näkyy, kuinka julkaisu prosessi etenee. Toisessa kuvassa (kuva 15) ei tarvitse muokata mitään ja tarvitsee vain painaa "Publish".



The screenshot shows the 'Publish' dialog box with the 'Connection' tab selected. The title bar reads 'Publish'. The main title is 'BuildaDogBot - Web Deploy'. The left sidebar contains 'Profile', 'Connection', 'Settings', and 'Preview'. The 'Connection' tab is highlighted in blue. The main area contains the following fields and controls:

- Publish method:** A dropdown menu set to 'Web Deploy'.
- Server:** A text box containing 'buildadogbot.scm.azurewebsites.net:443'.
- Site name:** A text box containing 'BuildaDogBot'.
- User name:** A text box containing '\$BuildaDogBot'.
- Password:** A text box filled with dots, with a 'Save password' checkbox checked below it.
- Destination URL:** A text box containing 'http://buildadogbot.azurewebsites.net'.
- Buttons:** 'Validate Connection', '< Prev', 'Next >', 'Publish', and 'Close'.

Kuva 14. Projektin julkaisu Install Wizard



The screenshot shows the 'Publish' dialog box with the 'Settings' tab selected. The title bar reads 'Publish'. The main title is 'BuildaDogBot - Web Deploy'. The left sidebar contains 'Profile', 'Connection', 'Settings', and 'Preview'. The 'Settings' tab is highlighted in blue. The main area contains the following fields and controls:

- Configuration:** A dropdown menu set to 'Release'.
- File Publish Options:** A section with a collapsed arrow icon.
- Databases:** A section with a message: 'No databases found in the project'.
- Buttons:** '< Prev', 'Next >', 'Publish', and 'Close'.

Kuva 15. Projektin julkaisu Install Wizard

Install Wizardin jälkeen Saa botin ID avaimen ja salasanan joita botti käyttää tunnistautumiseen Azuressa niin että sitä voidaan päivittää. Käyttäjätunnus ja salasana laitetaan toteutuksen Web.config-tiedostoon kuten alla olevassa kuvassa (Kuva 16) näkyy.

```
<configuration>
  <appSettings>
    <add key="BotId" value="YourBotId" />
    <add key="MicrosoftAppId" value="4dc08549-9e46-4ab1-a1b2-b6b86bc1dadd" />
    <add key="MicrosoftAppPassword" value="XXXXXXXXXXXXXXXXXXXX" />
  </appSettings>
</configuration>
```


Kuva 16. Botin ID ja salasana Web.config tiedostossa

Tämän jälkeen botin julkaisu Azureen on valmis.

5.3.3 Botin rekisteröiminen Microsoftin Bot Directoryyn

Kun botti on saatu ladattua Azureen, täytyy se vielä rekisteröidä Microsoftin Bot Framework sivun Bot Directoryyn, jotta pääsee muokkaamaan sen käyttämiä kanavia. Alla olevassa kuvassa (kuva 17) näkyy valmis tyhjä kaava, johon botin tiedot laitetaan.

Bot profile



Icon
Upload custom icon
30K max, png only

Name: * ?

Bot handle: * ?

Description: * ?

Configuration

Messaging endpoint:

Register your bot with Microsoft to generate a new App ID and password

[Create Microsoft App ID and password](#)

Paste your app ID below to continue

Admin

Owners: ?

Instrumentation key: ?

Kuva 17. Bot Directoryn rekisteröinti sivu

Sivu täytetään muuten normaalisti, mutta messaging endpoint kohta sivun puolella välissä tarvitsee botin Azure-osoitteen sekä botin app ID:n jonka saa Azureen rekisteröinnin yhteydessä.

Kun tämä kaavake on täytetty, tarvitsee vain rekisteröidä botti ja pääsee määrittämään sille kanavia Channels kohdasta sivun keskivaiheilla (kuva 18). Oletuksena botille on luotu web chat kanava jota voi kokeilla kuvassa oikealla näkyvässä keskustelu ikkunassa. Tämän lisäksi botille luodaan myös automaattinen Skype-kanava, jos bottia haluaa käyttää sitä kautta.

Jos haluaa botin voi vielä julkaista kaikkien näkyville, mutta itse en sitä tehnyt koska botti on vain prototyyppi.

Create a Dog Bot Publish

Details Edit

Bot handle
CreateaDogBot

Bot Framework Version
3.0

Messaging endpoint
<https://buildadogbot.azurewebsites.net/api/messag...>

Microsoft App ID
4dc08549-9e46-4ab1-a1b2-b6b86bc1dadd

Test connection to your bot

Test

Accepted

Skype and Web chat channels have been auto configured.
Click 'Edit' to fine tune channel specific capabilities and settings.

Channels

	Test link	Issues	Enabled	Published	
Direct Line		0	Yes	<input type="checkbox"/> Off	Edit
Skype	Add to Skype	0	Yes	<input type="checkbox"/> Off	Edit
Web Chat		0	Yes	<input type="checkbox"/> Off	Edit

[Get bot embed codes](#)

Add another channel

Email	Add
-------	------------------

Chat

Type your message...

Kuva 18. Botin ylläpito sivusto Microsoft Bot Directoryssa

6 JOHTOPÄÄTÖKSIÄ

6.1 Botin rakentamisen haasteet

Create a Dog botin rakentaminen ei ollut kovin vaikea projekti itsessään, mutta sehän olikin korvaava aihe ensimmäisen botin eli lukujärjestys botin epäonnistuttua. Lukujärjestys-botin rakentamisessa ongelmiksi muodostuivat puutteellinen ymmärrys C# ja jsonin välisestä toiminnasta ja siitä kuinka vaikeaa se on toteuttaa.

Dog botissa isompia haasteita itsessään ei ollut sillä sen tavoitteena oli tutkia olisiko tällaisen botin teettäminen opiskelijoilla hyödyllinen ja toteutettavissa oleva tehtävä.

Projektissa tuli opittua paljon bottien toiminnasta ja Microsoftin luomista palveluista sen ympärillä. Tärkeä oppitunti oli myös yhden botin epäonnistuminen, sillä se sai ymmärtämään, että aiheen valinnassa kannattaa olla hyvin tarkkana ettei yritä haukata liian isoa palaa taikka asiaa jota ei täysin ymmärrä.

6.2 Botin rakentaminen opiskelijan toimesta tehtävänä

Mielestäni botin rakentaminen on tehtävä sopiva opiskelijalle ja tarjoaa myös hyvin vaihtelua. Microsoft tarjoaa useita erilaisia pohjia joita käyttää bottien luomiseen, kuten hälytys-botti jolla käyttäjä voi luoda botin, joka laittaa viestiä jostain kiinnostavasta aiheesta taikka samankaltaisen FormFlow botin kuten itse tein.

Ainut iso kritiikki ja mahdollinen ongelma botin luomisena tehtävänä on Azuren maksullisuus. Tietenkin botti voitaisiin vain rajoittaa käyttämään vain Bot Framework Simulatoria, jolloin ei tarvita Azurea ollenkaan. Suositeltavaa kuitenkin olisi, jos koulu voi vaikka pyrkiä tekemään sopimuksen Microsoftin kanssa esim. kokeilu version käyttämisestä.

Koulun kannattaisi harkita botteja tehtävien kehittämistä, aluksi ihan Microsoftin omien esimerkkien varaan ja halutessaan kehittää haastavampia tehtäviä niitten ulkopuolelta. Azureen lataaminen ja Bot Directoryn käyttö eivät ole pakollisia, koska botteja voidaan emuloida simulaattorilla. Yksi vaihtoehto voisi olla että ensin opiskelijat harjoittelisivat yksinkertaisilla esimerkki boteilla ja sitten ryhmätyönä rakentaisivat monimutkaisempia botteja, jotka voitaisiin julkaista.

LÄHTEET

Bayerque, N. (2016). A short history of chatbots and artificial intelligence, 15.8.2016. Haettu 10.1.2017 osoitteesta:

<http://venturebeat.com/2016/08/15/a-short-history-of-chatbots-and-artificial-intelligence/>

Bisson.S. (2016). Build chat bots with Microsoft's Bot Framework. 21.12.2016. Haettu 10.1.2017 osoitteesta:

<http://www.infoworld.com/article/3152158/application-development/build-chat-bots-with-microsofts-bot-framework.html>

Férée, H. (2011). Turing Test. Haettu 10.1.2017 osoitteesta: https://commons.wikimedia.org/wiki/File:Turing_Test_Version_3.svg

Godara, A. (2016). A brief history of bots. 28.11.2016. Haettu 10.1.2017 osoitteesta:

<https://chatbotmagazine.com/a-brief-history-of-bots-9c45fc9b8901#.6ubhxp91>

Lavonen, Meisalo, Luovan ongelmanratkaisun työtavat

<http://www.edu.helsinki.fi/malu/kirjasto/lor/>

Microsoft (2016) Microsoft Bot Framework

<https://dev.botframework.com/>

Microsoft (2016) Language Understanding Intelligent Service (LUIS)

<https://www.microsoft.com/cognitive-services/en-us/language-understanding-intelligent-service-luis>

psych.fullerton.edu (1999)

<http://psych.fullerton.edu/mbirnbaum/psych101/Eliza.htm>

Wikimedia Commons (2007). SmarterChild. Haettu 10.1.2017 osoitteesta:

<https://commons.wikimedia.org/wiki/File:Pidgin-2.0.0b7-winxpro-conv.png>

Wotawa,F. Klassische Themen der Computerwissenschaften Artificial intelligence. Graz University of Technology

http://www.ist.tugraz.at/_attach/Publish/Ktdcw14/KTCW2006_Part1.pdf

Rodrigues, A. A History of SmarterChild. 16.3.2016. Haettu 10.1.2017 osoitteesta:

<http://motherboard.vice.com/read/a-history-of-smarterchild>

