Section 2: Introduction to Web Services

# 1. What is a Web Service?

Δεν είναι όλα τα Web Applications και Web Services. Για παράδειγμα, αν έχουμε δημιουργήσει ένα web app που διαχειρίζεται to-do-tasks, χρησιμοποιώντας MVC dependencies της Spring, τότε δεν έχουμε ένα web service. Αντιθέτως, θα είχαμε ένα εάν αφήναμε στην άκρη το View (δηλαδή την html, και επικεντρωνόμασταν στα Data και ο Business Logic και είχαμε ένα service το οποίο μπορεί να είναι επαναχρησιμοποιούμενο.

**Ορισμός Web Service**: Software System designed to support interoperable machine-to-machine interaction over a network.

**Τα τρία βασικά χαρακτηριστικά:**

* Designed for machine-to-machine (or application-to-application) interaction.
* Should be interoperable – Not platform dependent.
* Should allow communication over a network.

# Important How Questions related to Web Services

1. **How does data exchange between applications take place?**

Έχουμε ένα Application A, το οποίο καταλανώνει ένα Web Service. Το Application A στέλνει ένα input στο Web Service (που ονομάζεται request) ενώ το Service απαντάει με ένα output (που ονομάζεται response). Τα request και response είναι τα βασικά στοιχεία που αποτελούν ένα Web Service.

1. **How can we make we services platform independent?**

Σημαντικό στοιχείο για τα Web Services είναι να είναι ανεξάρτητα της πλατφόρμας στην οποία έχει δημιουργηθεί το Application. Για παράδειγμα ένα Web Service θα πρέπει να μπορεί να χρησιμοποιηθεί είτε από ένα Java Application, είτε από ένα .Net Application κτλ.

Οπότε το σημαντικό είναι ότι για να είναι το Service, ανεξάρτητο της πλατφόρμας, θα πρέπει και τα request/response να είναι ανεξάρτητα της πλατφόρμας. Θα πρέπει να είναι σε formats, που μπορούν να υποστηριχτούν από οποιοδήποτε πλατφόρμα.

Ένας τρόπος είναι τα XML (Extensible Markup Language).

Ένας δεύτερος τρόπος είναι τα JSON (JavaScript Object Notation), είναι ο τρόπος με τον οποίο η JavaScript παρουσιάζει τα αντικείμενα.

1. **How does the Application A know the format of Request and Response?**

Πως γνωρίζει το Application σε τι μορφή θα πρέπει να είναι το Request, καθώς και που να το στείλει; Η λύση σε αυτό το πρόβλημα είναι το Service Definition. Το οποίο καθορίζει:

1. Request/Response Format
2. Request Structure
3. Response Structure
4. Endpoint
5. **Web Services – Key Terminology**
   * **Request:** είναι η είσοδος του Service.
   * **Response:** είναι η έξοδος του Service.
   * **Message Exchange Format:** είναι το format του Request και του Response (XML/JSON).
   * **Service Provider:** Το Web Service που παρέχει το Service (π.χ. To-Do-Task Service).
   * **Service Consumer:** Το Application που καταναλώνει το Service (π.χ. Application A).
   * **Service Definition:** Είναι το «συμβόλαιο» ανάμεσα στον provider και των consumer, που καθορίζει τον τρόπο με τον οποίο θα γίνει η επικοινωνία.
   * **Transport:** Καθορίζει πως καλείται ένα Service (HTTP/MQ).

# Introduction to SOAP Web Services

Θα δούμε τους διαφορετικούς τύπους των Web Services: SOAP και REST.

Αξίζει να σημειώσουμε ότι το SOAP και το REST δεν είναι συγκρίσιμα. Το REST ορίζει μία αρχιτεκτονική προσέγγιση, ενώ το SOAP θέτει περιορισμούς στη μορφή των XML.

SOAP (Simple Object Access Protocol): Καθορίζει ένα συγκεκριμένο τρόπο για τη δημιουργία Web Services. Χρησιμοποιούμε XML ως exchange format.

# Introduction to RESTful Web Services

REST (Representational State Transfer), το κλειδί για τα REST Services είναι το γεγονός ότι θέλει να κάνει την καλύτερη χρήση του πρωτοκόλου HTTP.

Τι είναι το HTTP(Hyper Text Transfer Protocol) πρωτόκολο; Τα Requests και τα Responses είναι σε μορφή που ορίζεται από το HTTP. Όταν θέλω να κάνω ένα αίτημα σε ένα server με τον browser μου, τότε στέλνω στον server ένα GET request, ενώ ο server απαντάει με ένα HTTP response που περιέχει την HTML.

HTTP ορίζει μία κεφαλίδα (headers) και ένα body για το αίτημα μας. Επίσης μπορείς να δείξεις μέσα στο αίτημα και το είδος του αιτήματος, χρησιμοποιώντας τα request methods (π.χ. GET, POST, PUT κτλ.). Επίσης περιέχει και ένα status code (π.χ. 200, 404 κτλ.) όπου δείχνει την κατάσταση του αιτήματος (π.χ. ήταν επιχτυχημένο, error κτλ.)

Τα RESTful Web Services προσπαθούν να καθορίσουν τα services, χρησιμοποιώντας σενάρια που υπήρχαν ήδη (HTTP).

Ένα από τα βασικά χαρακτηριστικά των RESTful Web Services, είναι το Resource. Ως Resource θεωρούμε οτιδήποτε θέλουμε να «εκθέσουμε» στον έξω κόσμο μέσω του application.

Ένα resource αποτελείται από ένα URI και μπορεί να αναπαρίσταται με διάφορες μορφές όπως (XML, HTML, JSON).