Konark - Ad-Hoc Service Discovery

The Available APIs fall into these sections:

Registry APIs

The user takes two roles within the Konark protocol: either client or a server. When the user is looking for services, the device acts as a client. When one owns the service that a client wants to invoke, the device takes the role of a server. We will now take a look at the registry APIs for both a client and a server.

Client registry APIs

The client registry APIs exist in *Konark.clientservicemanager.ClientServiceManager*. This class includes the methods to get the available services discovered in the network. Also, there are methods to get the remaining time left to access these services and one can set what services to look for in discovery.

Konark.clientservicemanager.ClientServiceManager

1. ArrayList **getAvailableServices**() - This method gets all of the services that have been discovered by this device. All of the services are returned in the form of an ArrayList. One can then manipulate the services for display by casting each element in the ArrayList to AvailableSpecificService.

Konark.serverservicemanager.ServerServiceManager. These APIs are mainly for creating, managing, and deleting registered services.

Konark.serverservicemanager.ServerServiceManager

1.

Discovery APIs

Konark.serverservicemanager.ServerServiceManager

Konark.serverservicemanager.ServerAdvertisement

1.

ClientRequestHandler.ClientRequest+= - When this command is typed, the user will be prompted for the rest of it, hit "Tab" then hit "Tab" again and Visual Studio will automatically write the method signature for you. This event is fired to transfer control back to the main thread, so I would advice writing nothing in the body of the event method. However, one may do what they like with this event and may signal to the user that someone is attempting to access their services.

Client-side Service Discovery

These APIs are used by the client device to receive advertisements and to actively discover services using

discovery. There are two classes associated with these APIs.

Konark.clientservicemanager.ClientServiceManager

Konark.clientservicemanager.ClientDiscovery

1. void **discover**(string **pathInRegistry**) - This method spawns a new thread which runs another method. The second method sends a discovery message for all services in the specified path in the remote device's registry and waits for any replies. This method takes as a parameter the types of services it wishes to discover.

AdvertisedServiceHandler.ServiceHandler+= - When this command is typed, the user will be prompted for the rest of it, hit "Tab" then hit "Tab" again and Visual Studio will automatically write the method signature for you. You will need to fill in the body of the event handler method. When a service is discovered, an event is fired and can be caught by the main application or GUI. By filling in the even handler method, one can control what she wants to do with the new service.

Delivery APIs

HTTP Server for receiving Service Invoke

These APIs are used to start and stop the HTTP server on every Konark device. The server is responsible for handling requests for the full description of a service and finally the service invoke.

Konark.HTTP.Server.

1.

Client-side Service Invoker

These classes deal with methods to start and complete service delivery from the client's device.

Konark.deliverutils.ServiceInvoker

1.
