What do you think about breaking changes?

- Do you introduce them? If so, why?

What does the API do? [purpose of APIs]

- How do clients use the features of the API?
- How was the original design of the API thought out?
- How is the API documented?

Are changes made to APIs or are they set in stone? [changing an API -- policies]

- When do you decide to change an API?
- What decisions are made when changing an interface?
- Who is involved in the design/redesign of the API?
  - Is the client involved?
  - Are there multiple developer teams involved?
  - Do you employ an API/system architect who makes all the decisions?

How are API artifact changes introduced? [changing an API -- implementation & clients]

- Do you as the developer have any control over the upgrade behavior of your clients?
- What other ways can a release be made?
- Is there backward compatibility for older/obsolete features?
  - Steer the conversation to deprecation as a mechanism to achieve this
- Are the older features always completely replaced?

[deprecation]

- Do you use it?
- What do you think about it?
- How is your experience with it?
- Is there something you would improve about it?

Do you take any steps to lower the impact on the clients? [easing client burden]

- Does it happen that client code breaks after upgrading to a new version of the API?
- Are there any steps that you feel that would lessen the burden to upgrade?
- Do you document your changes thoroughly?
- Would deprecating features be a way to do so?

When a new feature is introduced (one that replaces an old one) would you prefer that clients transition to that one? [API developer recommendation]

- Does it lessen your burden if all clients are on the latest version of the API?
- How do incentivize developers to use the latest version of the API?
- If you were to use deprecation, do you think it will disincentivize users from transitioning?